

# Agile Mythbusting

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

Mary Ann Lapham  
Eileen Wrubel

Jan 2015



Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE <b>16 JAN 2015</b>		2. REPORT TYPE <b>N/A</b>		3. DATES COVERED	
4. TITLE AND SUBTITLE <b>Agile Mythbusting</b>				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) <b>Wrubel /Eileen</b>				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) <b>Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213</b>				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT <b>Approved for public release, distribution unlimited.</b>					
13. SUPPLEMENTARY NOTES <b>The original document contains color images.</b>					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT <b>SAR</b>	18. NUMBER OF PAGES <b>117</b>	19a. NAME OF RESPONSIBLE PERSON
a. REPORT <b>unclassified</b>	b. ABSTRACT <b>unclassified</b>	c. THIS PAGE <b>unclassified</b>			

**This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.**

**Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.**

**References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.**

**NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.**

**This material has been approved for public release and unlimited distribution except as restricted below.**

**This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).**

**DM-0002085**



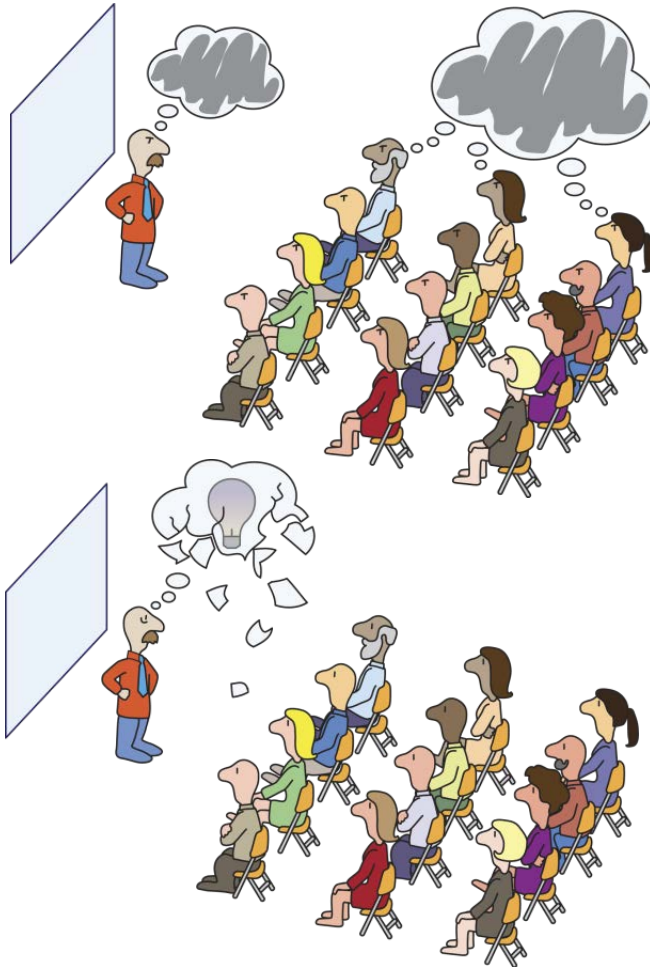
# Tutorial Goals

After completing this tutorial, participants will be able to:

- Describe common Agile myths for government settings and their source
- Debunk these common Agile myths
- List common Agile approaches seen in government settings
- Discuss common challenges to Agile adoption in any setting
- Discuss challenges to Agile adoption particular to regulated settings like the DoD
- Recognize potential Agile variants that are/are not productive in regulated settings



# Audience—Who are You?



*What role do you  
have in software  
acquisition?*

***Engineer***

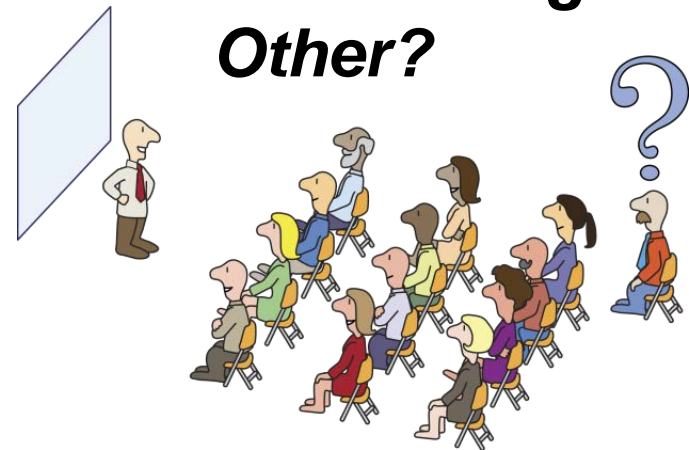
***Developer***

***Project Mgmt***

***Budget Staff***

***Contracting Staff***

***Other?***



# Tutorial Content

Why do we need Agile or lean software methods anyway?

Key Components of Agile Development: Agile principles

Top 10 Myths of Agile in DoD/government settings

Traditional and Agile Acquisition Life Cycles: Fixed vs evolving vision

Common Agile Methods: One size does not fit all

Scrum: The most adopted Agile method


Scaling Agile Methods: Going beyond the team level methods

Challenges to Agile Adoption: What's special about agile adoption in the DoD?

*BONUS: Gimmies and Gotchas in Agile adoption (time permitting)*



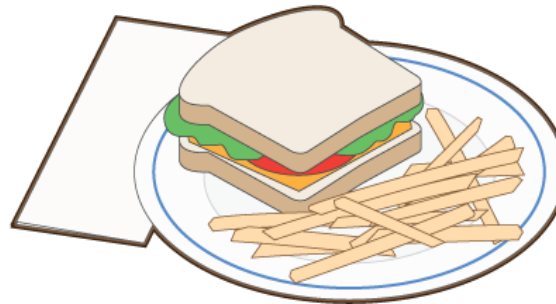
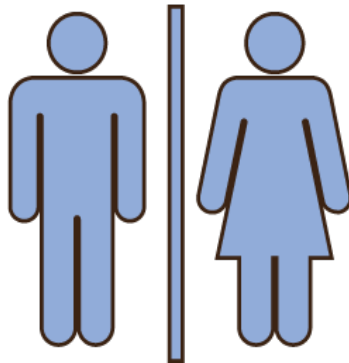
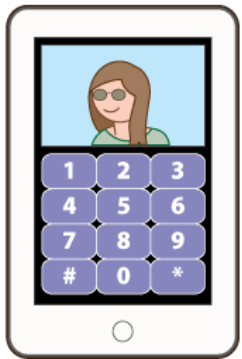
# What Else do You Need to Learn?

 Break into pairs and discuss what one thing BEYOND what we've said will be included that you want to get out of the tutorial (3 min)

Instructor will solicit answers and comment on what's in/what's out of scope for this tutorial.



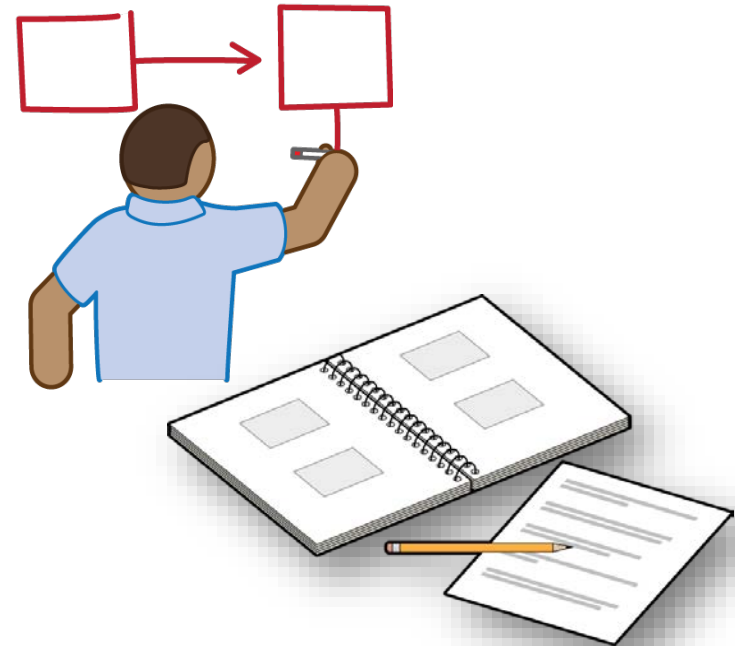
# Tutorial Logistics





# Course Approach

Exercises, discussions, presentations...



# Summary

This is an overview—we will touch on many topics, but rarely will have time to go into depth

We will do our best to integrate your learning needs into our plan

We welcome your sharing your experiences with Agile – if we cut you short, it's because of lack of time, not lack of interest



# WHY DO WE NEED AGILE SW DEVELOPMENT METHODS ANYWAY?

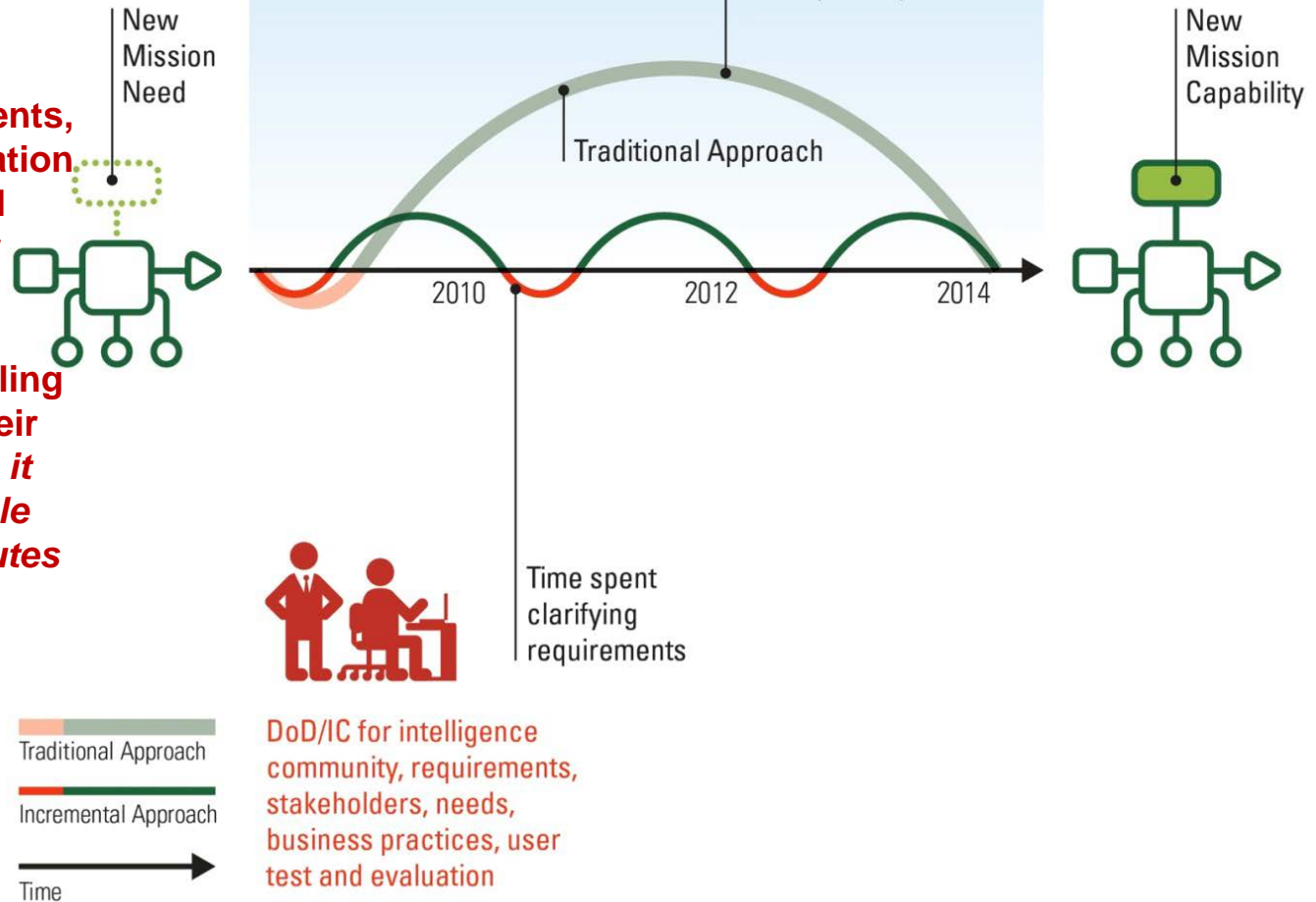


# DoD Acquisition and Innovation

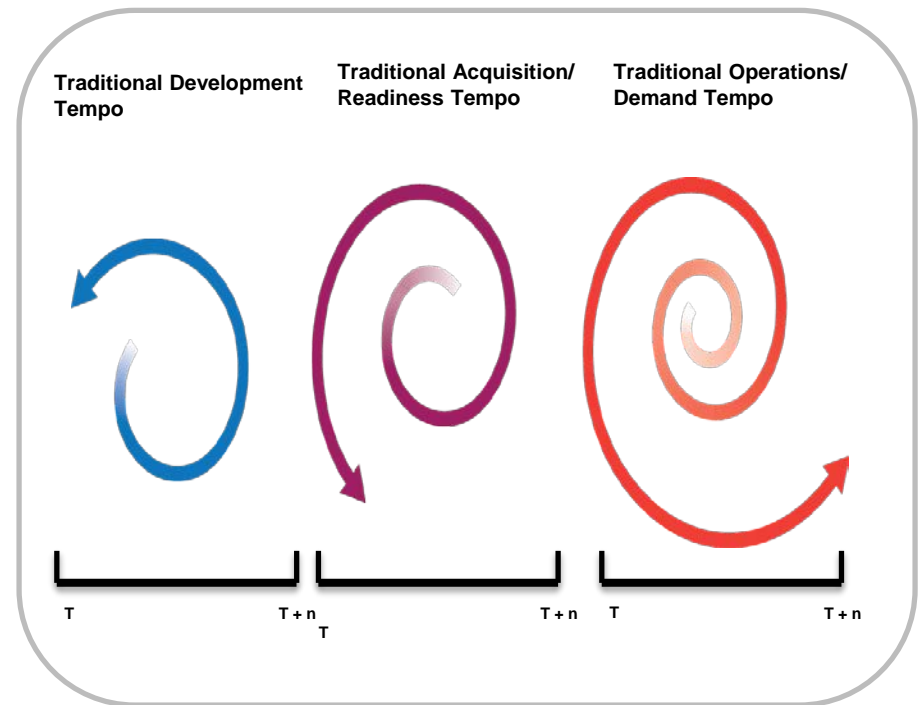
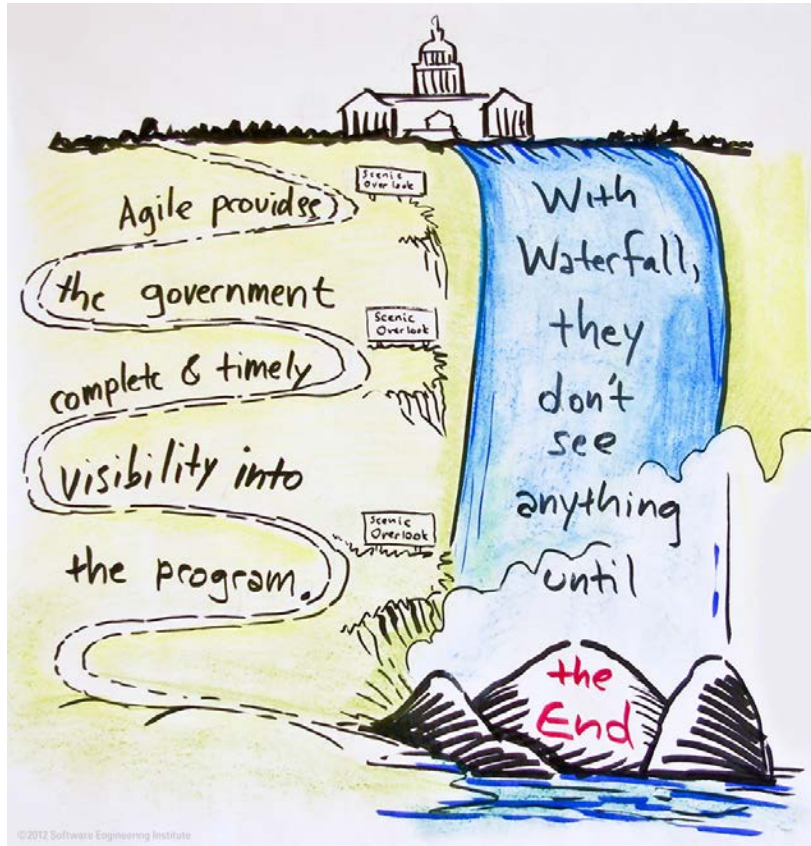
Systems and Software Engineering  
Expertise and Framework

Many regulated environments, like the DoD, NEED innovation and NEED incremental improvements to their systems.

Many of them are now willing to consider changing their approach if they can do it without getting in trouble with their governing statutes and regulations.



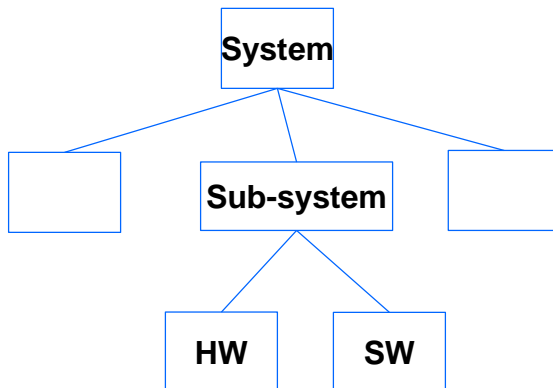
# The View of Our Customers



# Can software development be managed the same as hardware development?

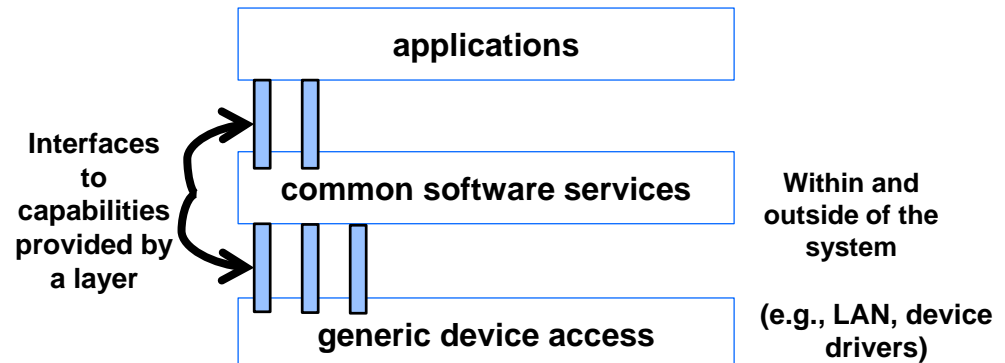
## Hardware Sys Development Assumptions

- Systems can be decomposed into discrete, independent, and hierarchically-related components (or subsystems).
- **Is part of:** Components can be constructed and integrated with minimal effort based on the original decomposition.
- Quality attributes can be allocated to specific components.



## Software Realities

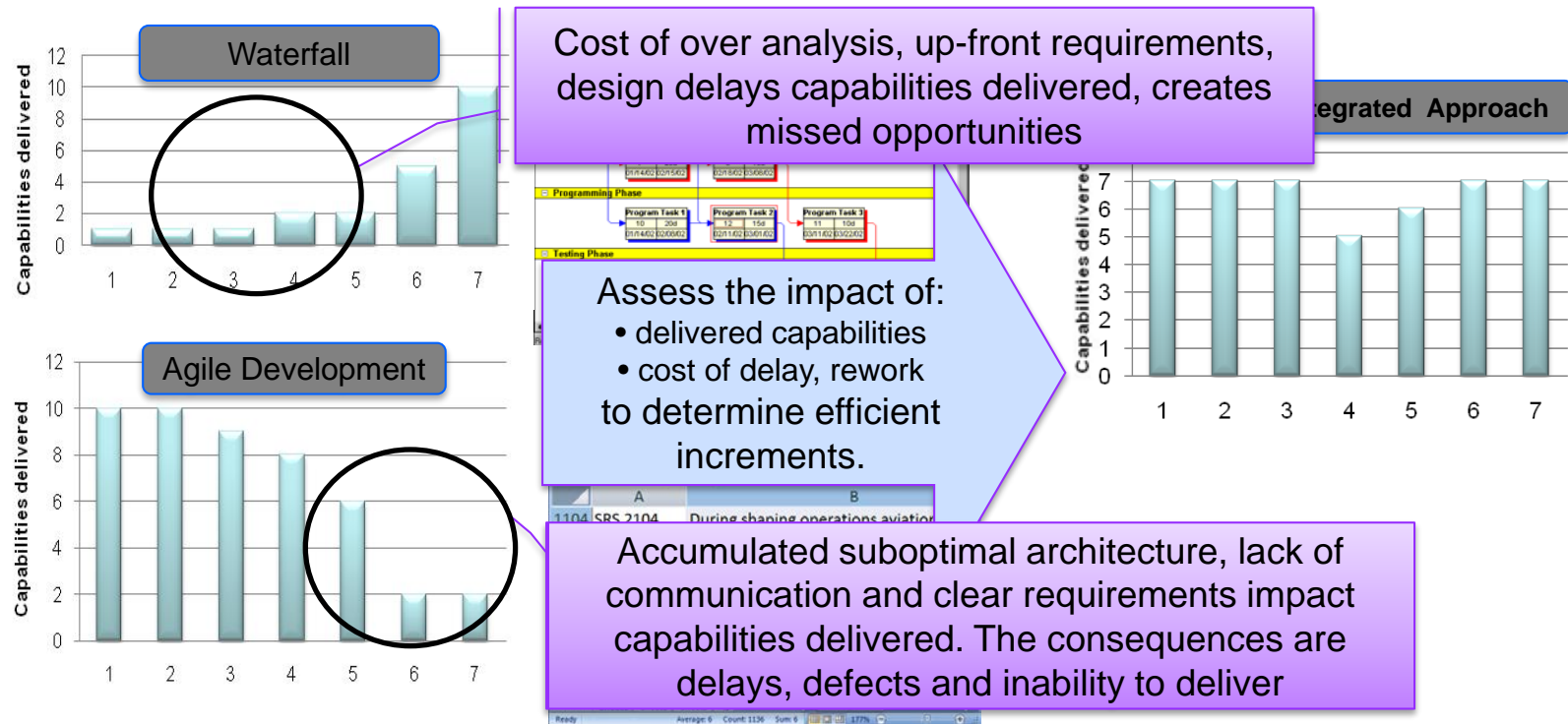
- Software components are often related sets of layered functionality (one layer is *not* contained inside another layer).
- **Is used by:** Interactions of the components (*not* the decomposition) must be managed.
- Quality attributes relate to composite interactions (*not* to individual components).



# Some Things Are Easier in Software



# Both Waterfall (HW-centric) and Agile Development (SW-centric) Have Risks



Ozkaya, Ipek. *Internal SEI customer presentation*, 2012.





# Historical Reasons SW Acquisitions Fail

Top 10 Reasons	Your Perspective
10. Technology used is new to the organization	
9. Software issues are considered too late in the system-development process	
8. Inadequate planning and estimating; long duration programs	
7. Size matters – large projects get into trouble more frequently than smaller ones	
6. Software objectives/requirements are not fully understood or specified; they change frequently (and grow) during the project; growth often uncontrolled/mismanaged	
5. Inadequate project management methodology	
4. Inadequate process emphasis	
3. Inadequate contract incentives to encourage use of modern software engineering practices	
2. Acquirers and developers lack experience working as a team	
1. Insufficient senior staff and/or inexperienced software engineering cadre	

Source: Nielsen, P. *Congressional Testimony* July 9, 2009.



# Summary

No single engineering life cycle or acquisition approach is perfect “out of the box” for individual acquisition situations

- Understanding the range of acquisition life cycle possibilities is a key to enabling a successful acquisition (using Agile or other approaches!)
  - Recent interim DoD 5000.02 guidance increases the explicitly-described variations of acquisition life cycles available to software – good news for those needing to use Agile methods

Software acquisitions have been failing for decades<sup>1</sup>

- This is a multi-factor problem with lots of complexity
  - Differences in what works for hardware-centric and what works for software-centric acquisitions is a common theme
- Some of the top failure modes have been mitigated in individual instances by robust Agile implementations
  - Duplicating their success means understanding Agile methods and their implications beyond tutorial level!

<sup>1</sup>See SEI’s “Acquisition Dynamics” research for other causes beyond what is discussed here See [www.sei.cmu.edu/acquisition/research](http://www.sei.cmu.edu/acquisition/research) for more details.



# KEY COMPONENTS OF AGILE DEVELOPMENT



# Agile Manifesto-Foundation of Agile Software Development

Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right,  
we value the items on **the left more**.

<http://www.agilemanifesto.org/>

**Common myth:**  
**The manifesto is often misinterpreted to mean**  
**no documentation,**  
**no process, and**  
**no plan!**



# Agile Principles Accompanying the Manifesto <sub>1</sub> –

## All are important aspects of building an agile culture

1. Highest priority is satisfy the customer through early and continuous delivery of software.
2. Welcome changing requirements, even late in development...
3. Deliver working software frequently, from a couple of weeks to a couple of months...
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Provide environment and support they need...
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.



# Agile Principles Accompanying the Manifesto <sub>2</sub> –

## All are important aspects of building an agile culture

7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development...a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Adapted from <http://agilemanifesto.org/principles.html>



# Exercise



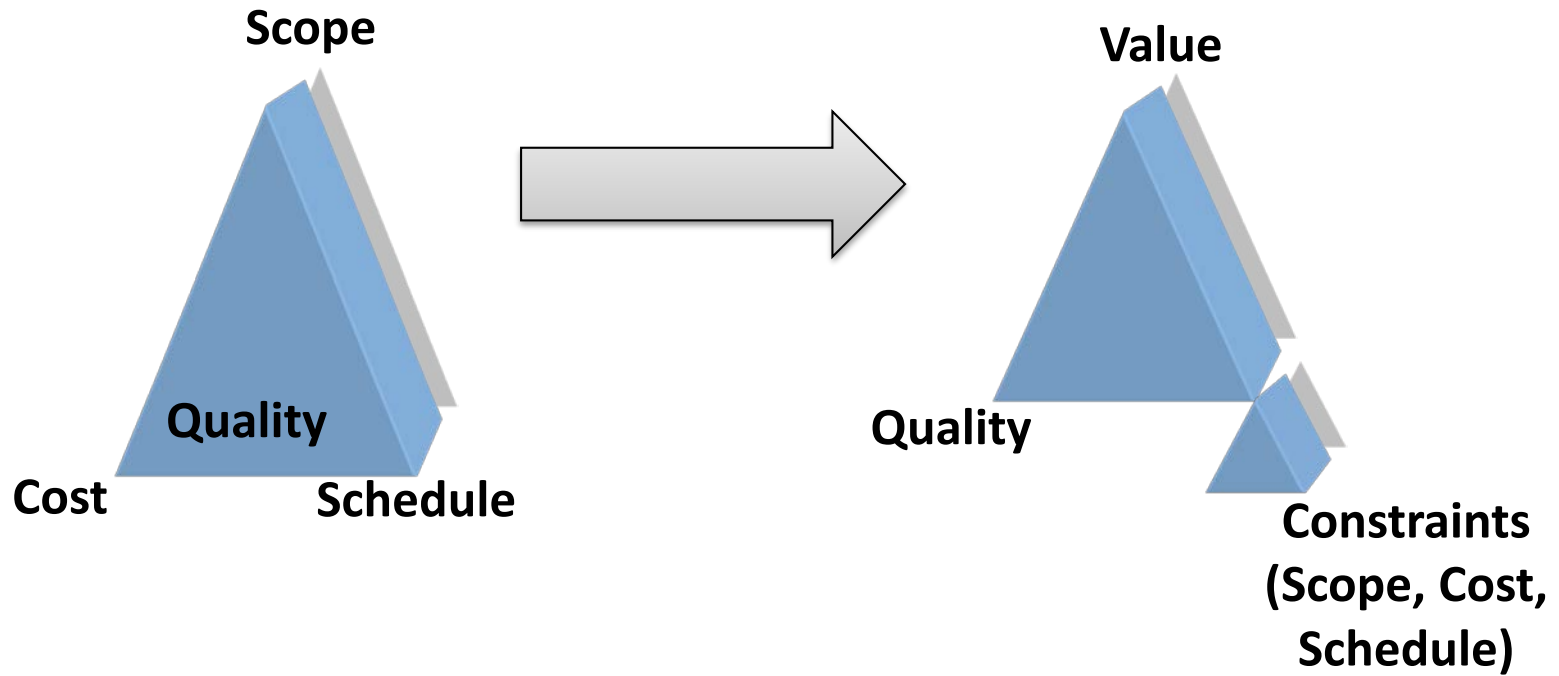
Break into pairs and discuss “What myths have you heard related to Agile that could come from the manifesto or principles?”

Each pair provides two sticky notes, each with one idea, to instructor's flip chart

Instructor will quickly group them and tell group which ones are addressed in this tutorial and which are not.



# Changing the Trade Space



*Adapted from Jim Highsmith (<http://www.jimhighsmith.com/2010/11/14/beyond-scope-schedule-and-cost-the-agile-triangle/>).*





# Some Common Characteristics of Agile implementations

**Iterative** —elements are expected to move from skeletal to completely fleshed out over time, not all in one step

**Incremental** —delivery doesn't occur all at once

**Collaborative** —progress is expected to be made by stakeholders and the development team working collaboratively throughout the development time frame

**Parallel** —multiple self-organizing, cross-functional teams work concurrently on multiple product elements (e.g., requirements, architecture, design, and the like for multiple loosely coupled product components)

**Dedicated** —team members are allowed to focus on the tasks within an iteration/release as opposed to multi-tasking across multiple projects

**Time-boxed** —relatively short-duration development cycles that permit changes in scope rather than changes in delivery time frame

Adapted Nidiffer, Miller, & Carney. *Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development & Management*, SEI-2013-TN-006



# Exercise



Using the handout provided, follow the facilitator's instructions carefully.



# Summary

The key to successful Agile implementation is understanding how you will instantiate the Agile manifesto and principles

When using Agile methods, the trade space changes – from fixed vision and evolving time, to fixed time and evolving vision

The Agile principles have implications for the characteristics of the life cycle that can be used

- But there's still more than one valid way of implementing the principles



(opinion of the authors, not an empirical survey)

# **TOP 10 AGILE MYTHS IN REGULATED SETTINGS**

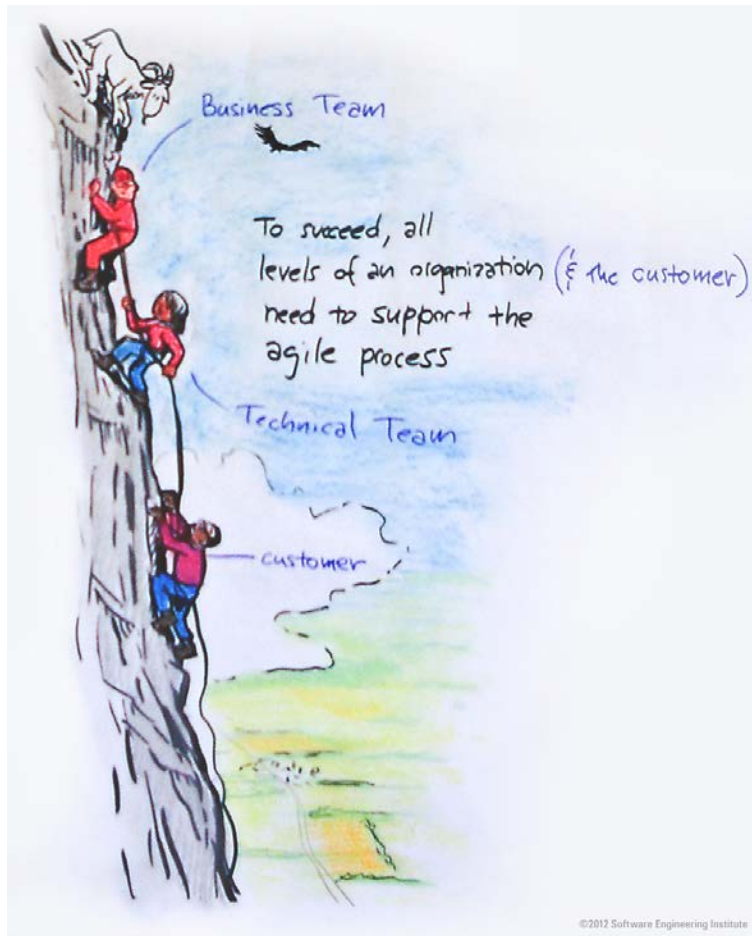


# Which myths do you most want to talk about?

	Your Vote
Agile is a fad—if I wait long enough, it will go away	
Agile teams don't document anything	
Agile is “cowboy” programming	
Agile only works in co-located environments	
Agile is just incremental, or spiral, or iterative, renamed	
Agile won't “stick” in DoD/regulated environments	
Agile only works with small projects	
Agile software teams can succeed in a vacuum	
You must choose Agile or Waterfall—you can't do both	
You can't use Earned Value Management on Agile Software Developments	



# Myth: Agile software teams can succeed in a vacuum



# Myth: Agile is a Fad...if I wait long enough, it will go away...

DoD and NDAA documents tend to suggest that DoD IT projects follow Agile-like processes and lifecycles

Federal working groups/task forces in place to support these directives (e.g. Section 804 Task Force) [AFE2012]

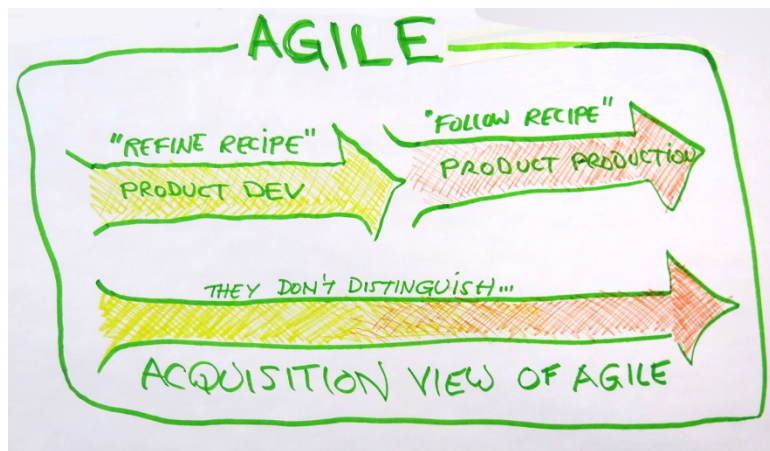
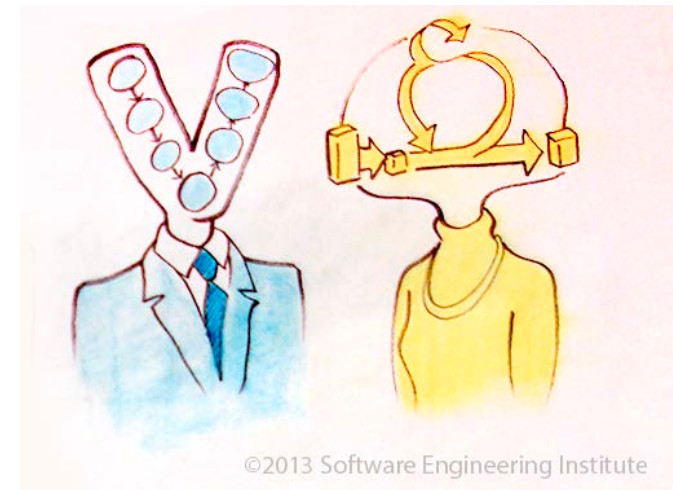
Government is looking at alternative development processes to enable earlier delivery of capability to users.

Interim DoD 5000.02 guidance include hybrid life cycle examples that more easily accommodate Agile methods implementation.



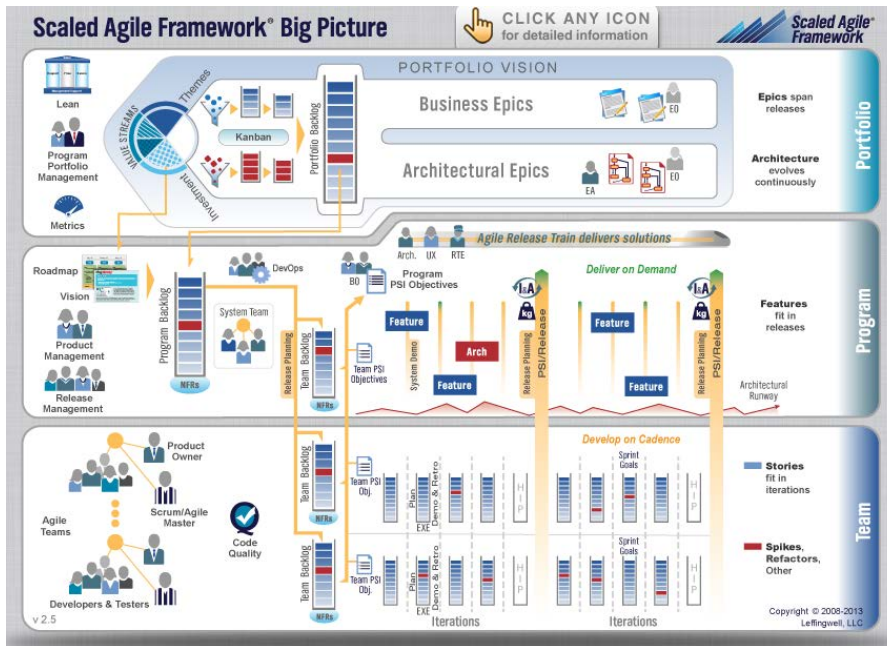
# Myth: You Must Choose Agile or Waterfall – you can't do both

What about “water-scrum-fall”?





# Myth: Agile only works for small projects....



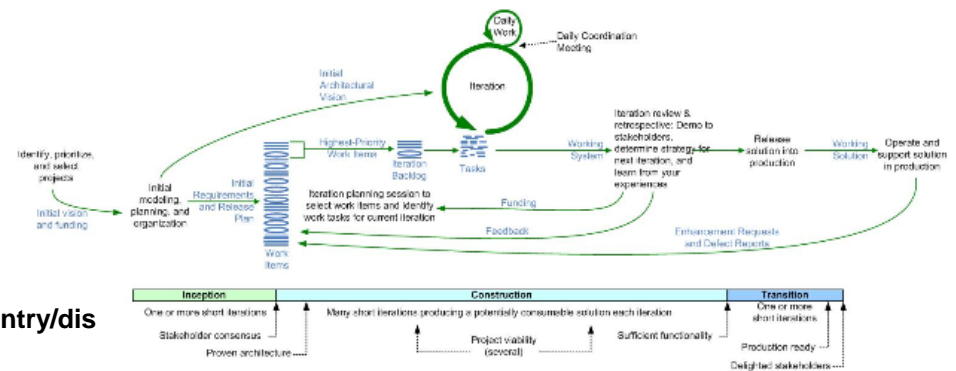
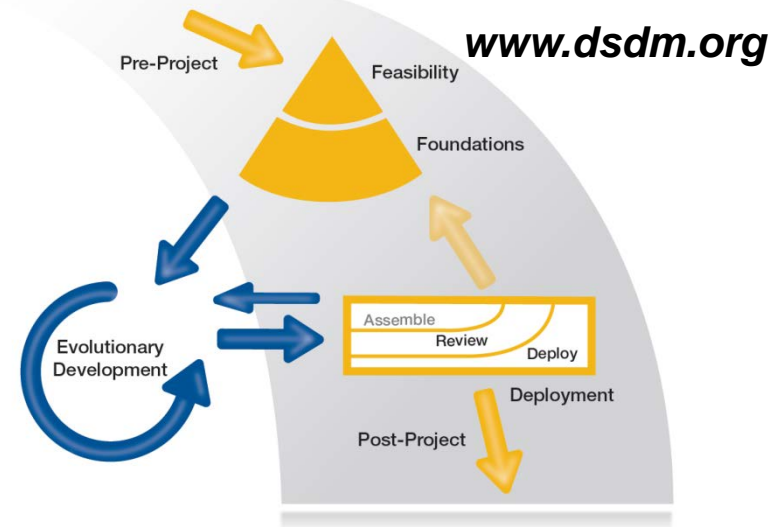
**SAFe**

[www.scaledagileframework.com](http://www.scaledagileframework.com)

**DAD**

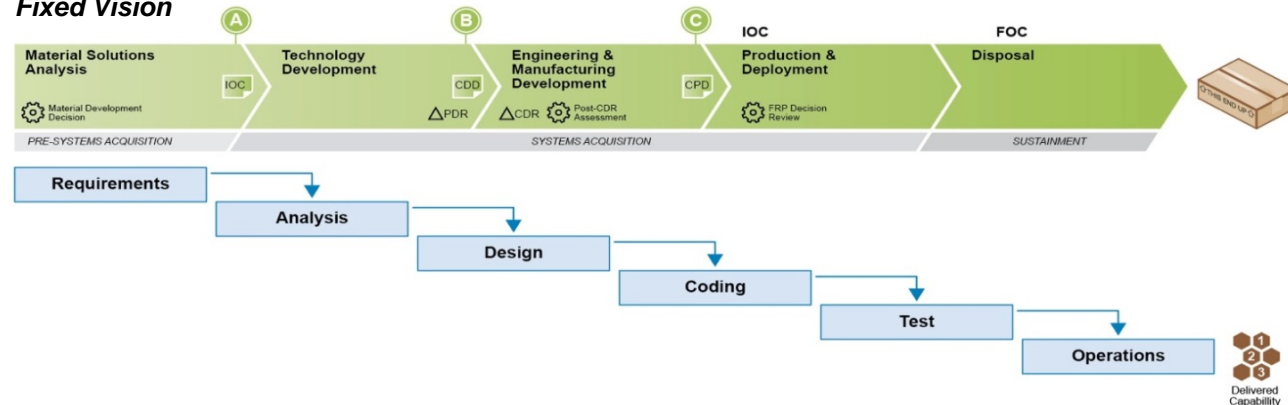
[https://www.ibm.com/developerworks/community/blogs/ambler/entry/disciplined\\_agile\\_delivery\\_dad\\_lifecycle14?lang=en](https://www.ibm.com/developerworks/community/blogs/ambler/entry/disciplined_agile_delivery_dad_lifecycle14?lang=en)

**DSDM**

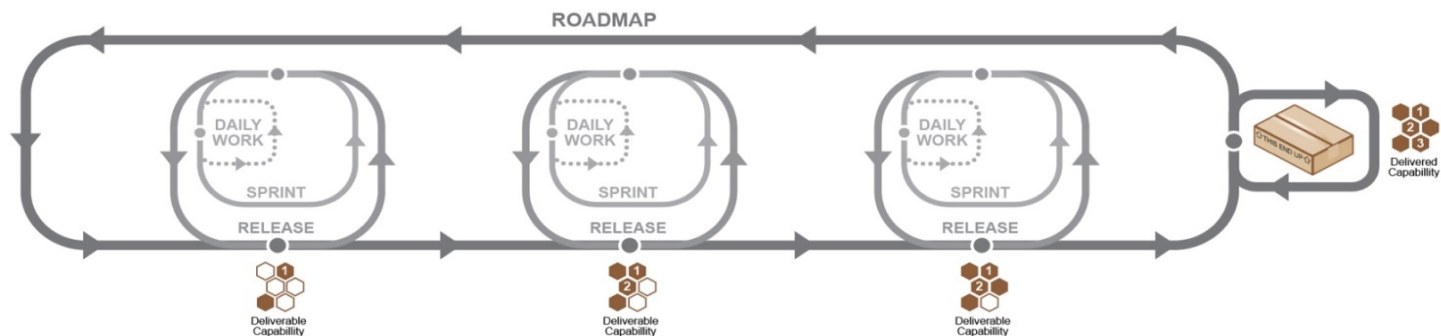


# Myth: Agile is just incremental, or spiral, or iterative, renamed

## Fixed Vision



## Evolving Vision



Source: Palmquist, Steven; Lapham, Mary Ann; Garcia-Miller, Suzanne; Chick, Timothy; & Ozkaya, Ipek. *Parallel Worlds: Agile and Waterfall Differences and Similarities* (CMU/SEI-2013-TN-021). Software Engineering Institute, Carnegie Mellon University, 2013.



# Myth: Agile Won't "Stick" in DoD Environments

It's a Journey...Patriot Excalibur switched to Agile methods in 2003 and successfully continues today



**The Agile Journey of Patriot Excalibur (PEX)**

A case of "Fortunate Serendipity"

- Reimbursable funding model
- Beneath the acquisition radar in the test wing

**Adopted XP in 2003**

- more SMEs
- more releases
- more functionality delivered

Went from 20 to 200+ adapters in ONE YEAR!

**2007 Agile Struggles**

- More demand led to a bigger team
- 20 on a team was too big!
- "traditional" smells crept in

THAT doesn't smell like the future!

**2008 Agile ReOrg 1**

- Quasi-Scrum model
- Independent teams
- More SMEs
- Add in SCM

★ MISSION REASSIGNED TO AFLCMC "Agile Evangelists"

- Accredited as a "software program" (like MS Word) instead of a system

**2009/10 Expansion**

- Demand increased to 600 adapters
- Increased # of teams ~ 100 people
- Architecture → SOA
- Geographically distributed team
- Added automated testing team

**2010 Difficulty Scaling**

- 7 week test event @ end of iterations
- teams still very independent
- introduced technical debt that built up

**2011 Agile ReOrg 2**

- Adopted Scrum "by the book"
- Product Owner Team
- Embedded Tester
- More automation
- Integrated Design Teams
- Stabilized architecture
- @ end of iteration

DEFINITION OF DONE

Red light Green light

**2012-2013 Scrum**

- Host @ 16 locations
- Every MATCOM uses it
- Moved to GSA T&M contract - SWSdev as a service
- Single code base still a key
- Added specialized skill of security
- Multiple technologies side-by-side: Web & Client Server ~ 2 million SLOC
- AFLCMC Support

**Open Issues**

- Architecture Committee is inefficient
- Long-range planning
- 7-week system test at the end
- Reliable funding stream
- Code quality & Test Coverage

From SEI Agile Collaboration Group Colloquium, March 2013. Used with permission.

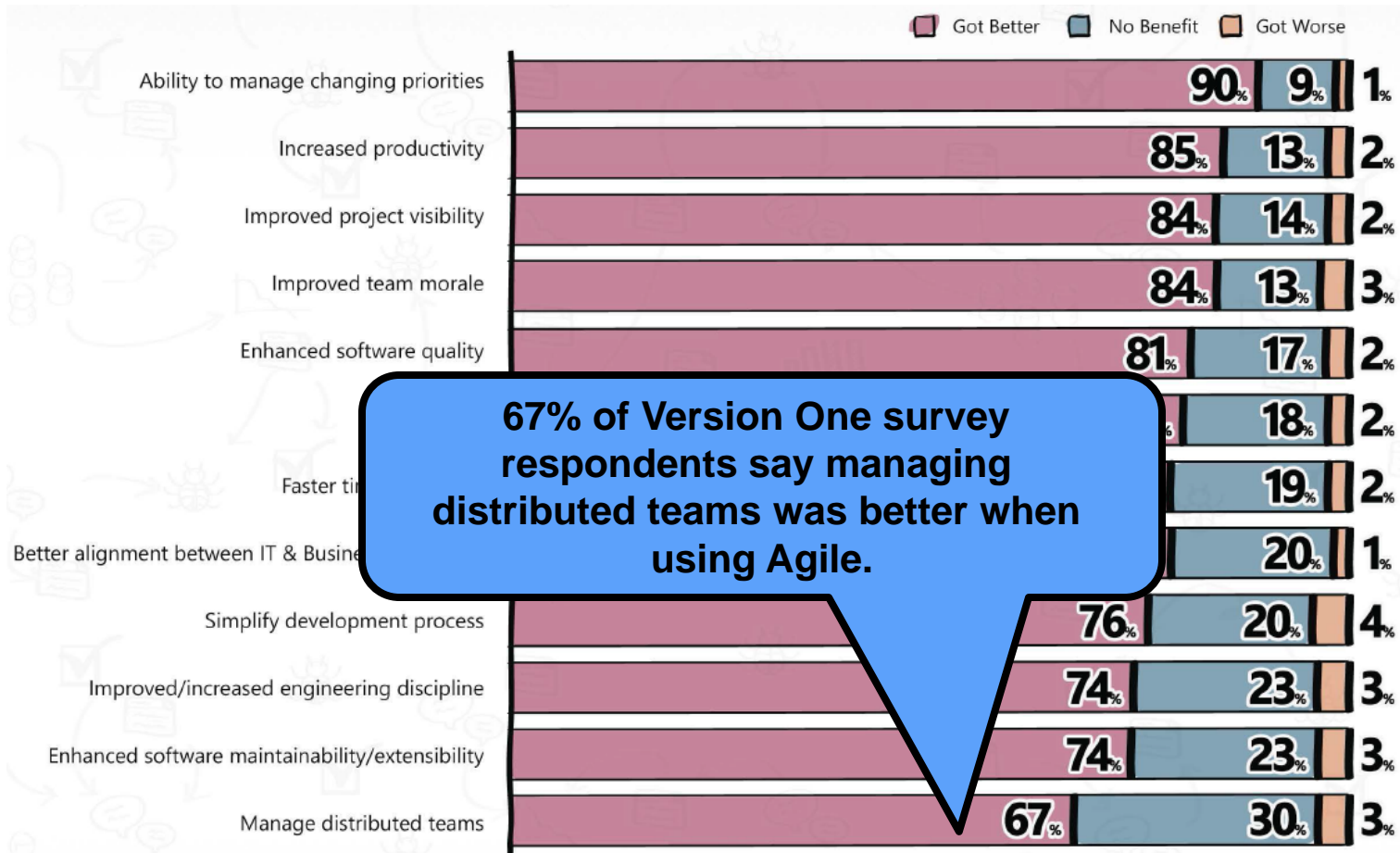
©2013 Software Engineering Institute

<http://www.crosstalkonline.org/storage/issue-archives/2004/200407/200407-Fortier-Lozancich.pdf>,

<http://www.dtic.mil/ndia/2011agile/NDIAAgileProcessinDoD.pdf>



# Myth: Agile Only Works in Co-Located Environments



# Myth: Agile is “cowboy programming”



Break into pairs and review the Agile principles -- which of those principles supports “cowboy programming” as described below?  
(description taken from “10 types of programmers you'll encounter in the field” by Justin James)

Be prepared to discuss your thoughts with the larger class.

## The Code Cowboy

The Code Cowboy is a force of nature that cannot be stopped. He or she is almost always a great programmer and can do work two or three times faster than anyone else. The problem is, at least half of that speed comes by cutting corners. The Code Cowboy feels that checking code into source control takes too long, storing configuration data outside of the code itself takes too long, communicating with anyone else takes too long... you get the idea.

The Code Cowboy's code is a spaghetti code mess, because he or she was working so quickly that the needed refactoring never happened. Chances are, seven pages' worth of core functionality looks like the "don't do this" example of a programming textbook, but it magically works. The Code Cowboy definitely does not play well with others. And if you put two Code Cowboys on the same project, it is guaranteed to fail, as they trample on each other's changes and shoot each other in the foot.

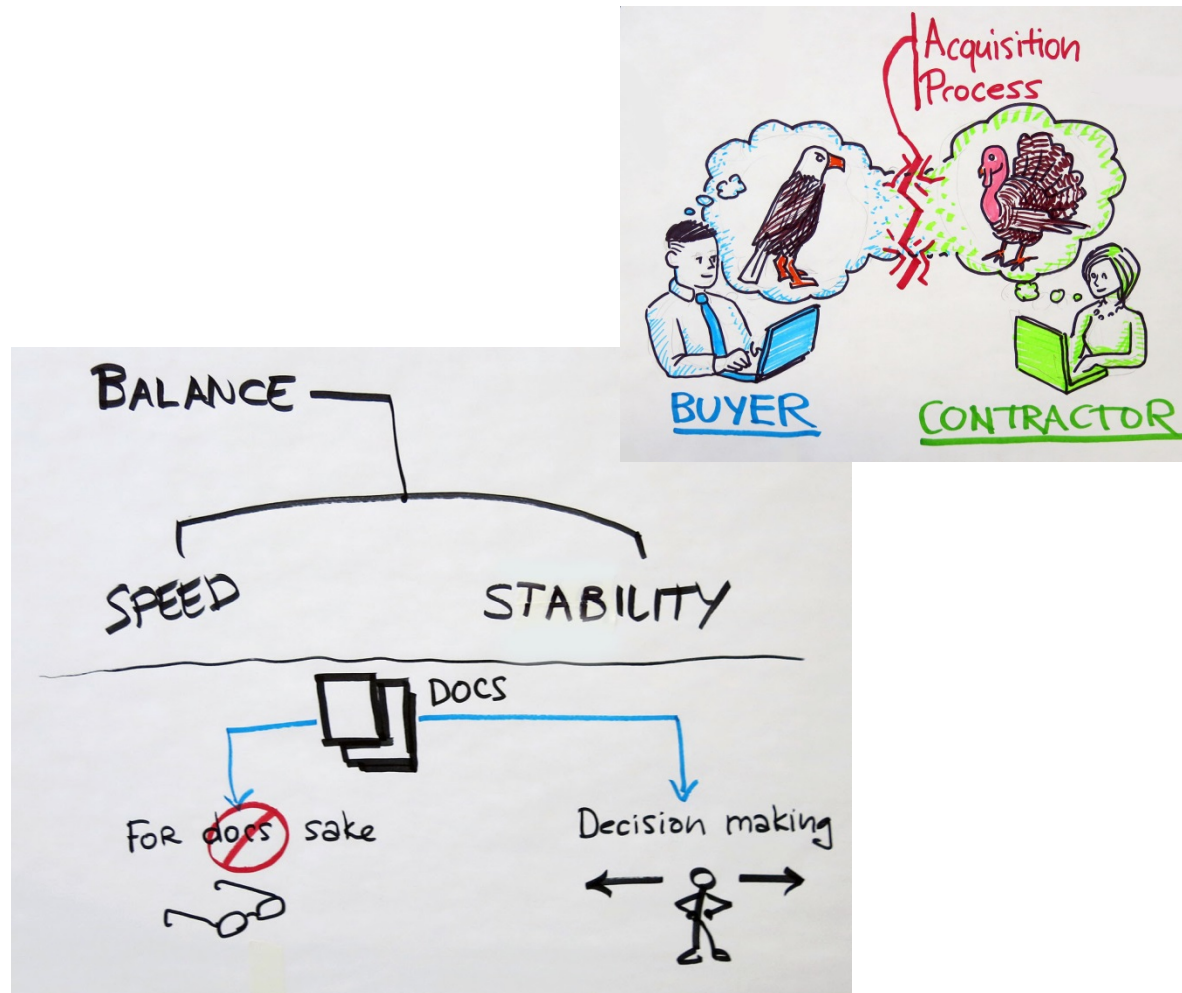
Put a Code Cowboy on a project where hitting the deadline is more important than doing it right, and the code will be done just before deadline every time. The Code Cowboy is really just a loud, boisterous version of The Ninja. While The Ninja executes with surgical precision, The Code Cowboy is a raging bull and will gore anything that gets in the way.

Source: <http://www.techrepublic.com/blog/10-things/10-types-of-programmers-youll-encounter-in-the-field/262/>

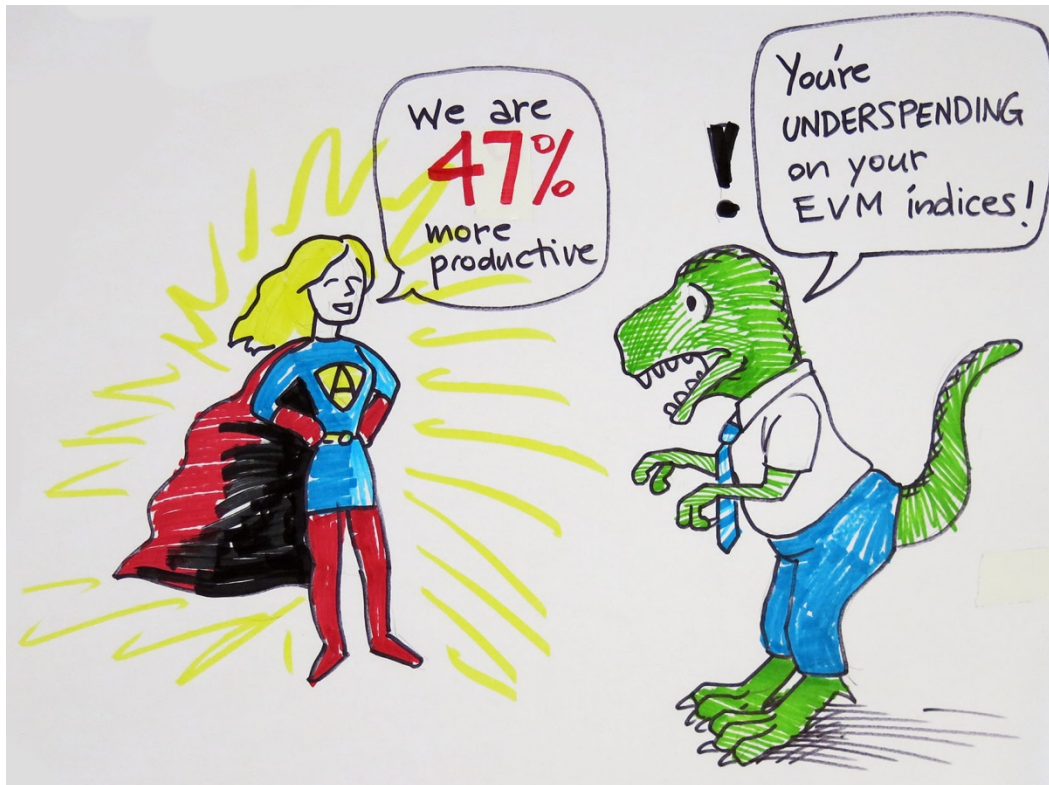




# Myth: Agile Teams don't Document Anything



# Myth: You Can't Use Earned Value Management with Agile Software Development



“Agile EVM” is successfully used in multiple environments, including DoD programs.<sup>1</sup>

<sup>1</sup>Start with “Agile EVM in Scrum Projects” from AGILE 2006 to get started learning about Agile & EVM.  
<http://www.computer.org/csdl/proceedings/agile/2006/2562/00/25620007-abs.html>



Top 10 Reasons	Your Perspective
10. Technology used is new to the organization	
9. Software issues are considered too late in the system-development process	
8. Inadequate planning and estimating; long duration programs	
7. Size matters – large projects get into trouble more frequently than smaller ones	
6. Software objectives/requirements are not fully understood or specified; they change frequently (and grow) during the project; growth often uncontrolled/mismanaged	
5. Inadequate project management methodology	
4. Inadequate process emphasis	
3. Inadequate contract incentives to encourage use of modern software engineering practices	
2. Acquirers and developers lack experience working as a team	
1. Insufficient senior staff and/or inexperienced software engineering cadre	

Source: Nielsen, P. Congressional Testimony July 9, 2009.



Break into pairs and discuss how following Agile principles might mitigate these historical SW acquisition failures?

Be prepared to discuss your thoughts with the larger class.





# Summary

Like all myths, Agile myths usually are based on some actual experience

- Sometimes anomalous
- Often happens when teams who are trying to mimic Agile practices without understanding Agile principles fail in some way

We have seen counterexamples FOR ALL THESE MYTHS in DoD and other government settings

- Agile isn't always the answer to a software acquisition's problems, but these myths shouldn't be a reason to avoid it

There are lots of other lists of Agile myths (see References at the end of the tutorial)

- We have focused on ones that we commonly hear about related to government settings



# TRADITIONAL & AGILE ACQUISITION LIFE CYCLES

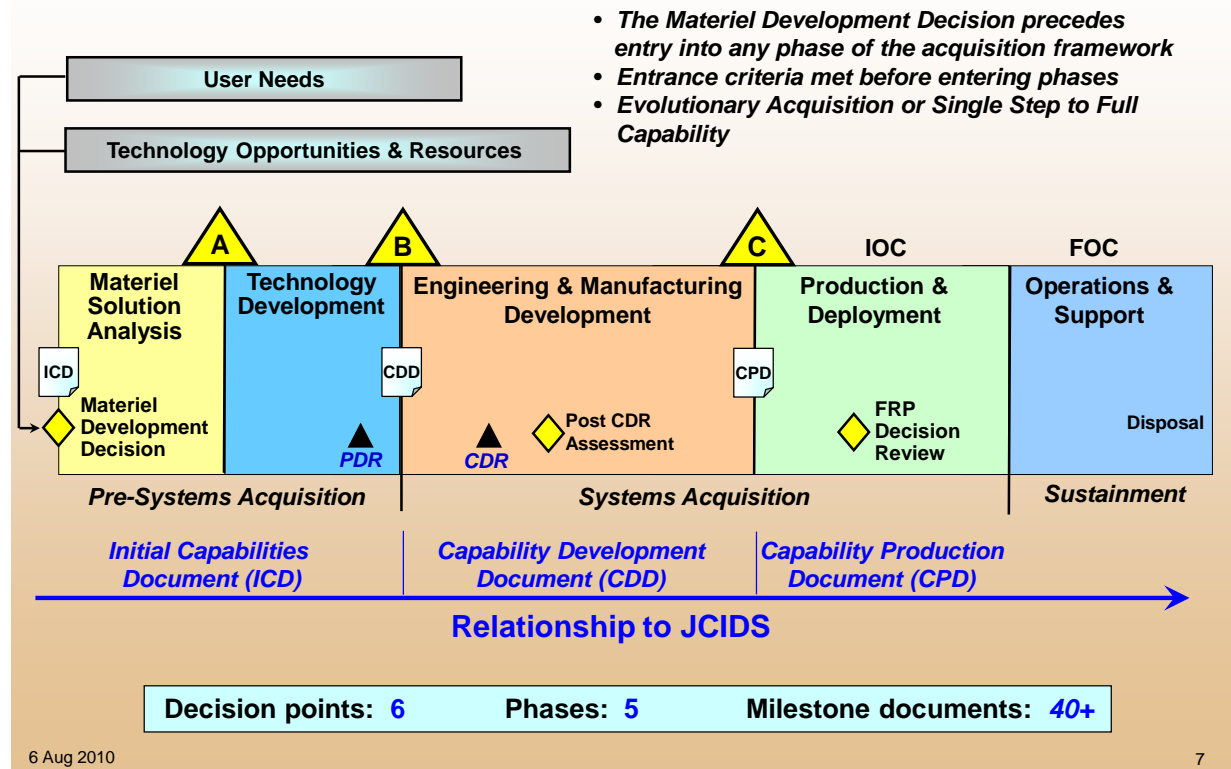


# Overarching Framework for System Acquisition

*Some things at the “next level down” have more explicit variety in the new DoD 5000.02 published in January 2015.*



## The Defense Acquisition Management System



6 Aug 2010

7

Source: Palmquist, Steve, et al. *Parallel Worlds*:



# New Life Cycle Descriptions in DoD 5000.02

Figure 8. Model 6: Hybrid Program B (Software Dominant)

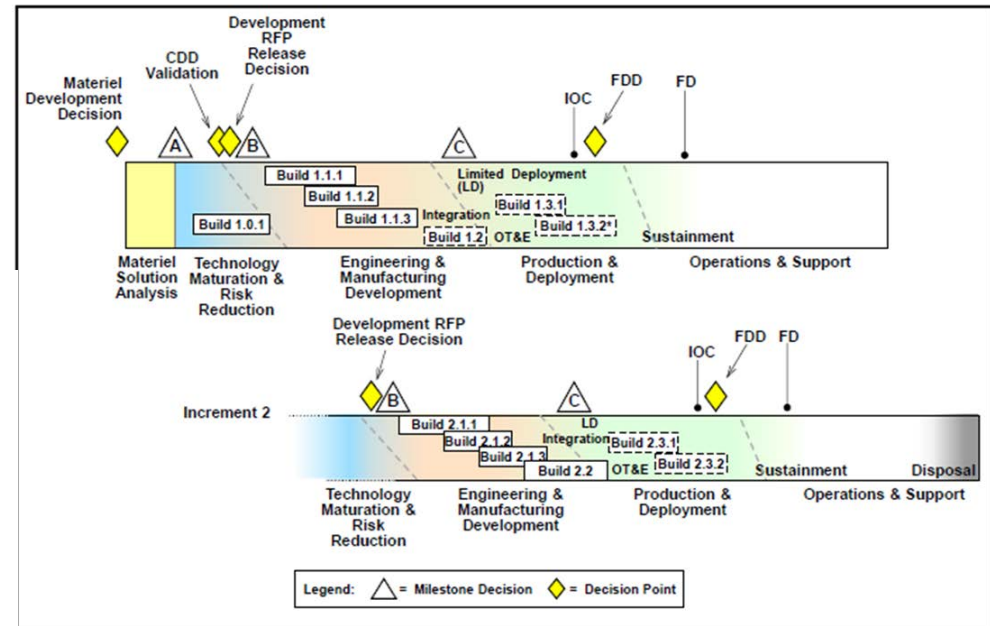
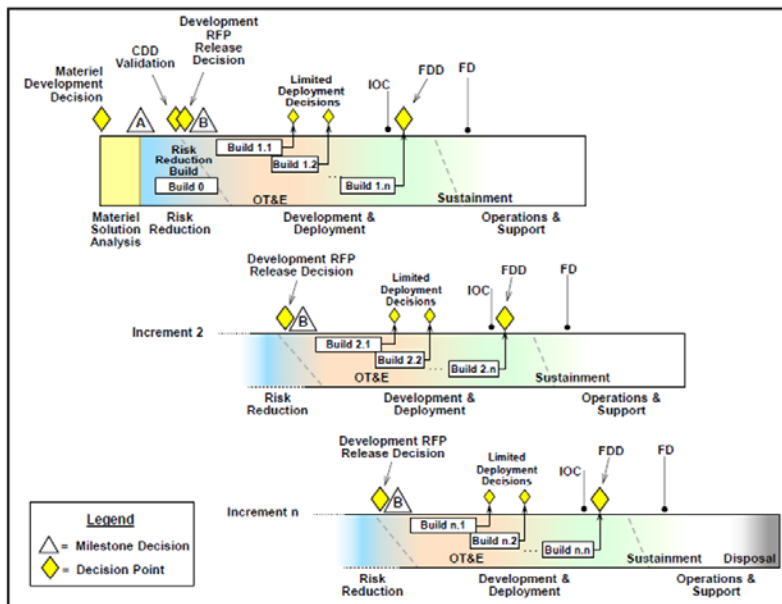


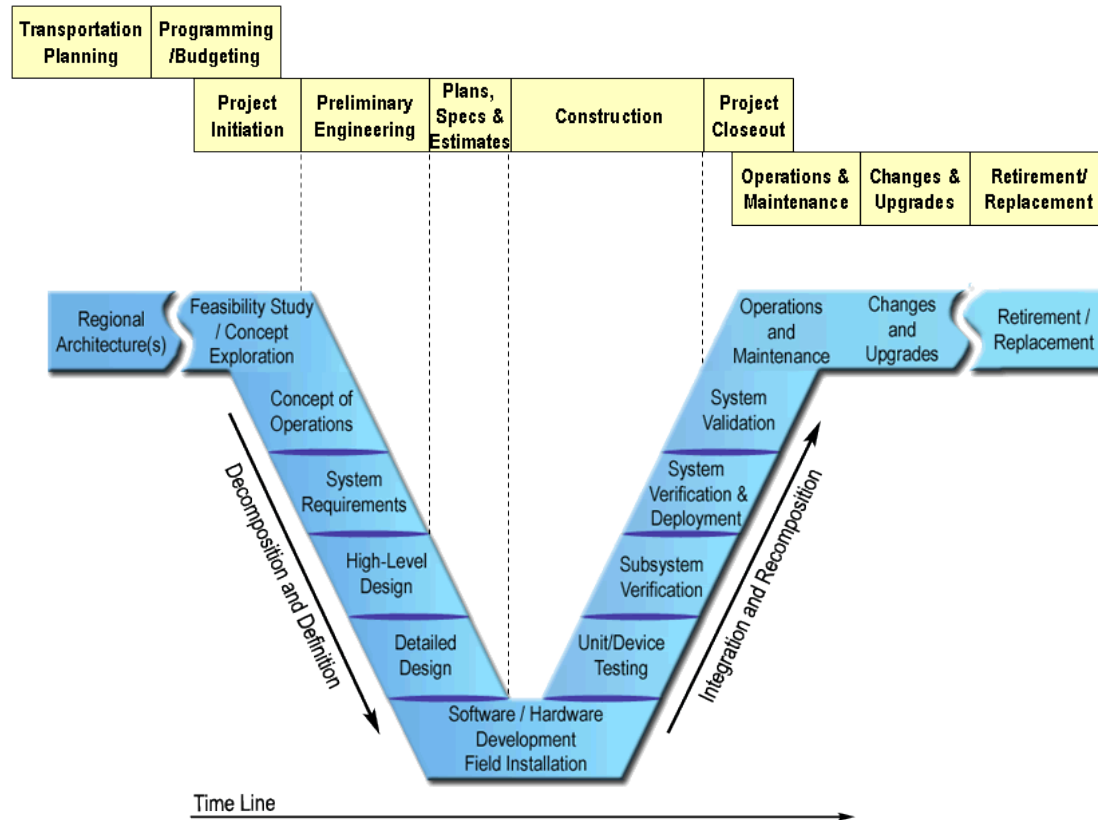
Figure 5. Model 3: Incrementally Deployed Software Intensive Program



<http://www.acq.osd.mil/docs/500002p.pdf>  
issued Jan 7, 2015



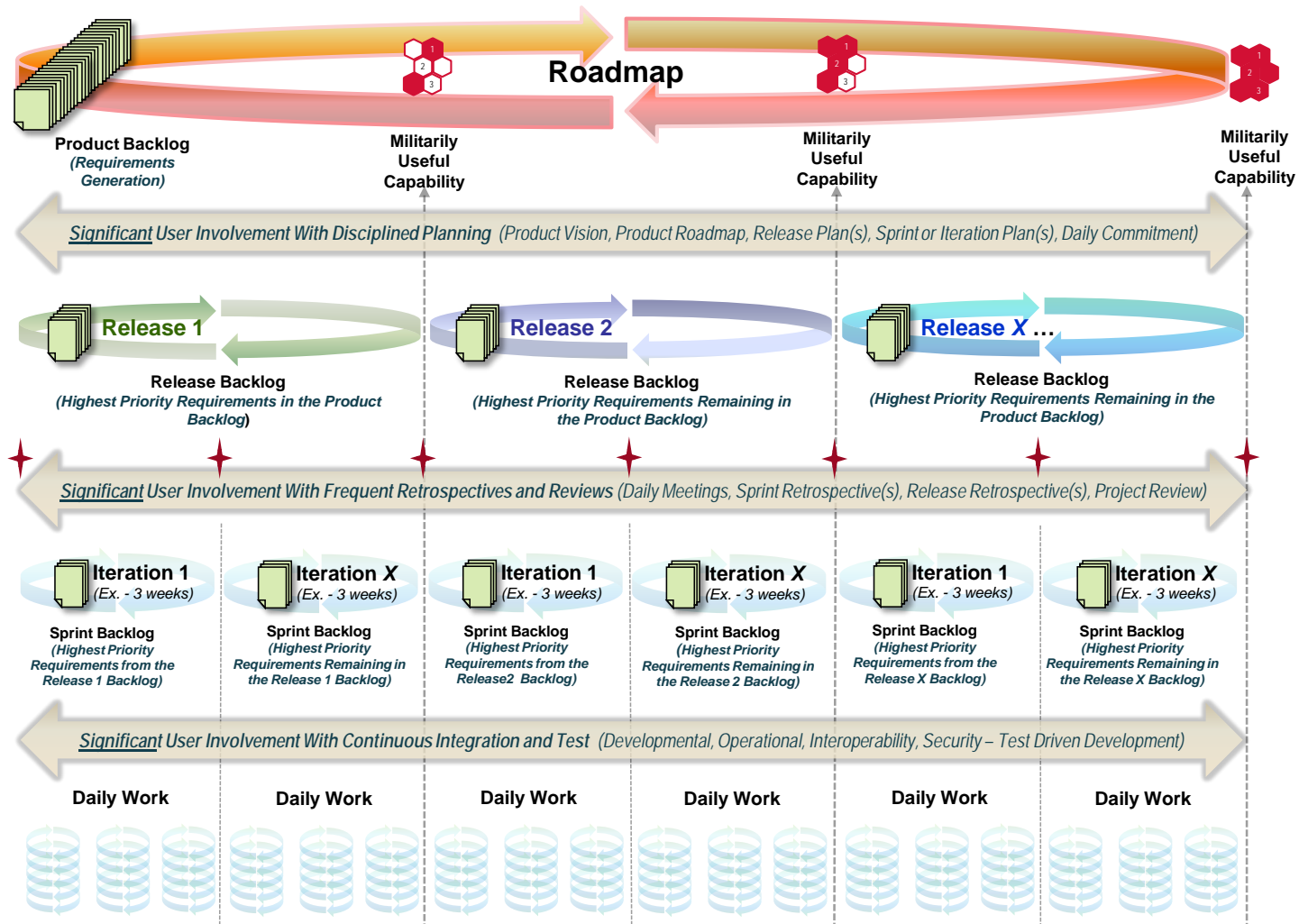
# The Classic Engineering “V Model”



Source: Palmquist, Steve, et al. *Parallel Worlds*:



# One View of Large Scale Agile Development

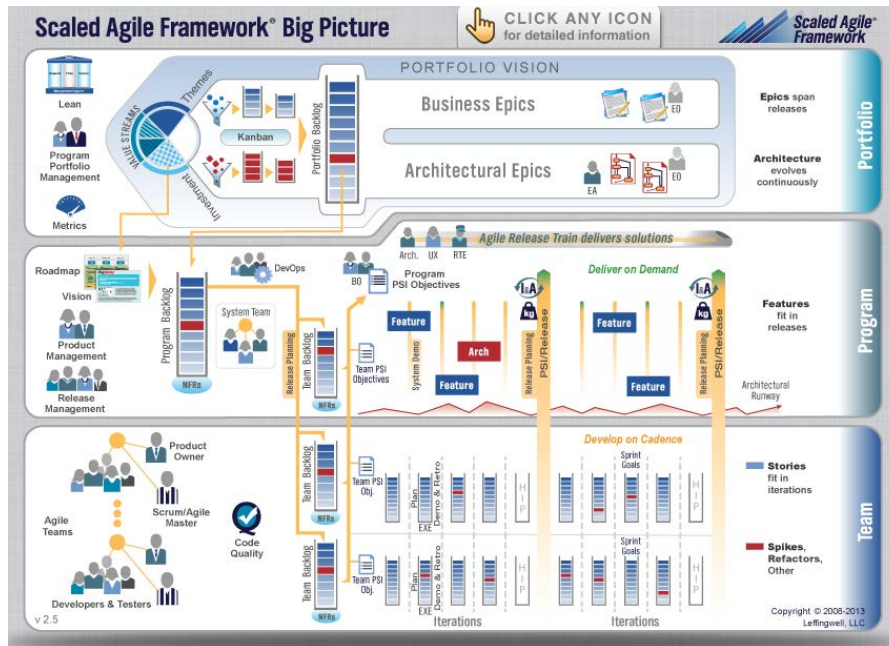


Source: Palmquist, Steve, et al. *Parallel Worlds*:

1



# Three Commercial Approaches to Agile at Scale



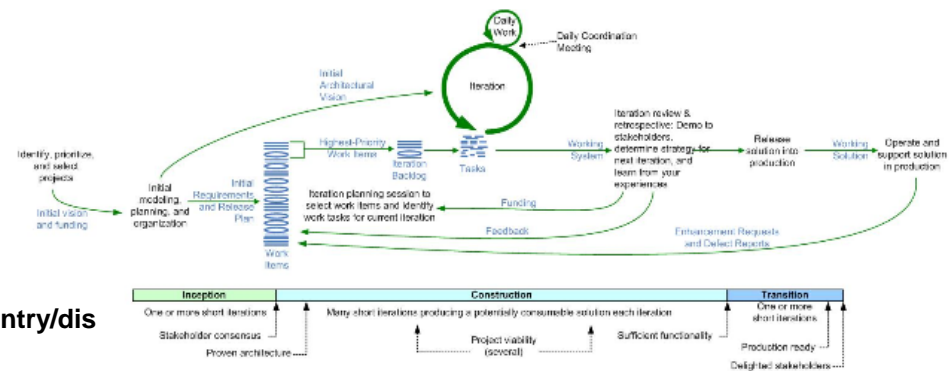
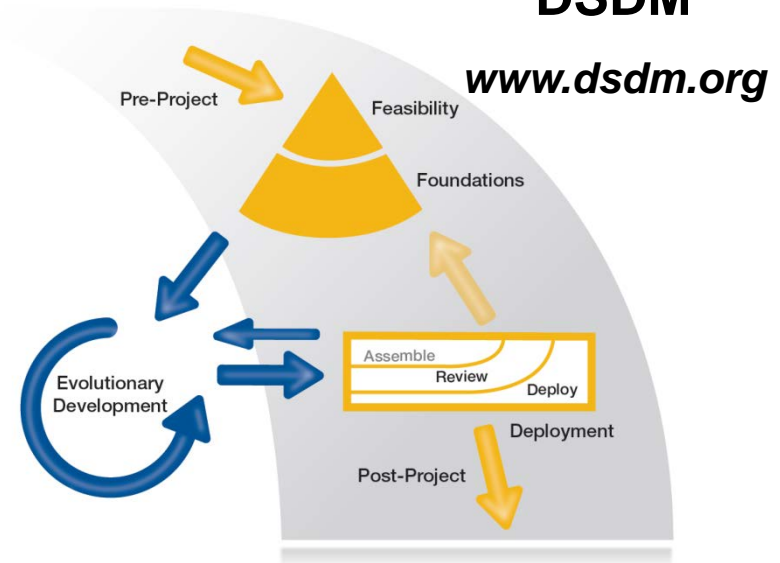
**SAFe**

[www.scaledagileframework.com](http://www.scaledagileframework.com)

**DAD**

[https://www.ibm.com/developerworks/community/blogs/ambler/entry/disciplined\\_agile\\_delivery\\_dad\\_lifecycle14?lang=en](https://www.ibm.com/developerworks/community/blogs/ambler/entry/disciplined_agile_delivery_dad_lifecycle14?lang=en)

**DSDM**



# Reconciling Traditional and Agile Life Cycle Viewpoints

Traditional at its Worst	Agile at its Best
Plan the work—especially the budget, schedule, and deliverables—to the maximum extent possible before beginning any design or code.	<ul style="list-style-type: none"> <li>• Near-term plans contain more detail, while plans further out on the time horizon contain fewer details.</li> <li>• The overall vision is broken down into a roadmap, which is further broken down into release plans, which are further broken down into sprint or iteration plans, which are further broken down into daily plans.</li> <li>• Requirements are prioritized.</li> <li>• Cost and schedule estimates are prepared for each capability at a high level. Relative estimation versus absolute estimation is employed.</li> <li>• Frequent planning sessions (at the beginning of each iteration) result in detailed, high-fidelity plans.</li> <li>• Risks are assessed and risk mitigation influences planning.</li> </ul>
Lock down requirements to prevent gold-plating and scope creep.	<ul style="list-style-type: none"> <li>• No requirements can be added to an iteration once it has started.</li> <li>• New requirements are evaluated by the stakeholders and prioritized thus preventing gold-plating and scope creep.</li> </ul>
Institute multiple reviews to provide senior leadership oversight as well as to serve as gates for continued work.	<ul style="list-style-type: none"> <li>• The customer is involved in all aspects of planning and testing. Customer (in the form of the product owner) is involved daily.</li> <li>• There are reviews at the end of each iteration that serve as gates to further work.</li> </ul>
Move forward in a step-by-step, sequential manner and only when all parts of the previous steps were complete.	<ul style="list-style-type: none"> <li>• The code base is integrated and tested daily.</li> <li>• The code base must pass all tests before and after integration. Regression testing is typically done each night.</li> </ul>
Capture all details with extensive documentation.	<ul style="list-style-type: none"> <li>• There is an overall plan.</li> <li>• There are requirements descriptions.</li> <li>• There are cost and schedule estimates.</li> <li>• There are risk assessments.</li> <li>• There is training material (as appropriate).</li> <li>• There is documentation (as appropriate).</li> <li>• There are lessons learned (based on retrospectives).</li> </ul>

Source: Palmquist, Steve, et al. *Parallel Worlds*:





# Traditional vs Agile Approaches Fit

## Traditional approach

- consistent with the acquisition life cycle guidance provided in the DoD Acquisition Deskbook and its supporting documents.
- programs with stable requirements and environment, with known solutions to the requirements
- programs with a homogeneous set of stakeholders who communicate well via documents
- programs for which the technology base is evolving slowly (technology is not expected to be refreshed/replaced within the timeframe of the initial development)

## Agile approach

- programs with volatile requirements and environment
- programs where solutions are sufficiently unknown that significant experimentation is likely to be needed
- programs for which the technology base is evolving rapidly
- programs with stakeholders who can engage with developers in ongoing, close collaboration

Nidiffer, K. Miller, S. & Carney, D. *Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development and Management* (CMU/SEI-2013-TN-006)



# Traditional approach

Strengths of the traditional approach include:

- enables the comparability and repeatability that standardization provides
- enables a contractually verifiable definition of completed intermediate work products
- reduces risks by means of contractually assured baselines

Weaknesses of the traditional approach include:

- the process drives measurement of compliance with itself as a primary measure of success (i.e., rather than measuring success as deploying a workable solution)
- it depends on documents as the basis to verify and validate the requirements, the architecture, and the detailed design
- most of the requirements are completed before any code is written, thus extending development timelines

Nidiffer, K. Miller, S. & Carney, D. *Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development and Management* (CMU/SEI-2013-TN-006).



# Agile approach

Strengths of this approach include

- early insight by the users into the shape of the solution
- early course correction
- “fail fast” (If the early solution ideas turn out to be flawed, little time or money is spent before that learning occurs.)
- explicit understanding that the requirements are expected to evolve

Weaknesses of this approach (particularly in large acquisition settings) include

- more dependence on tacit knowledge (e.g., lack of explicit documentation) as the basis for decision-making than is comfortable for most acquisition organizations
- dependence on availability of actively engaged user/customers
- difficulty in aligning implementation-driven artifacts and measures with those of the larger traditional acquisition setting.

Nidiffer, K. Miller, S. & Carney, D. *Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development and Management* (CMU/SEI-2013-TN-006).



# Summary

Interim DoD 5000.02 (November 2013) expands the explicit descriptions of life cycles expected to be used in DoD acquisitions

- Should make it easier for programs interested in Agile methods to adopt them within explicit 5000.02 guidance

## A Key Difference in Traditional and Agile Approaches

- Forward-Looking Perspective vs. Backward-Facing Perspective
  - In a dynamic environment like the DoD, the Traditional World struggles to deliver as it constantly looks back at long-fixed requirements and priorities.
  - In a dynamic environment like the DoD, the Agile World adapts as it delivers by constantly looking forward at evolving requirements and priorities.



# COMMON AGILE METHODS



# Methods Generally Termed “Agile”

## Scrum

- focused on team management practices

## XP (Extreme Programming)

- focused on team technical practices

## Dynamic Systems Development Method (DSDM)

- Popular in UK, based on RAD (Rapid Application Development)
- One of the oldest Agile methods, designed for large scale

## Crystal

- Encourages risk-based selection of practices; different patterns for different contexts

## Test Driven Development (TDD)

- Technical and management practices focused on writing the test that proves acceptance, then coding to that

## Disciplined Agile Delivery (DAD)

- Derived from Rational Unified Process, designed to scale

## Scaled Agile Framework (SAFe)

- Merger of lean and other Agile methods to support large scale projects



# All Agile Methods Share Most of These Characteristics

**Iterative** —elements are expected to move from skeletal to completely fleshed out over time, not all in one step

**Incremental** —delivery doesn't occur all at once

**Collaborative** —progress is expected to be made by stakeholders and the development team working collaboratively throughout the development time frame

**Parallel** —multiple self-organizing, cross-functional teams work concurrently on multiple product elements (e.g., requirements, architecture, design, and the like for multiple loosely coupled product components)

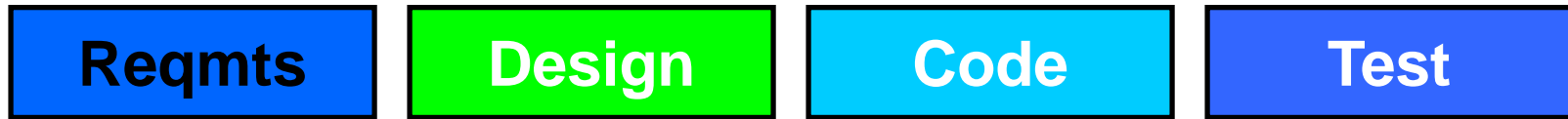
**Dedicated** —team members are allowed to focus on the tasks within an iteration/release as opposed to multi-tasking across multiple projects

**Time-boxed** —relatively short-duration development cycles that permit changes in scope rather than changes in delivery time frame

Adapted from Nidiffer, Miller, & Carney. *Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development & Management*, SEI-2013-TN-006

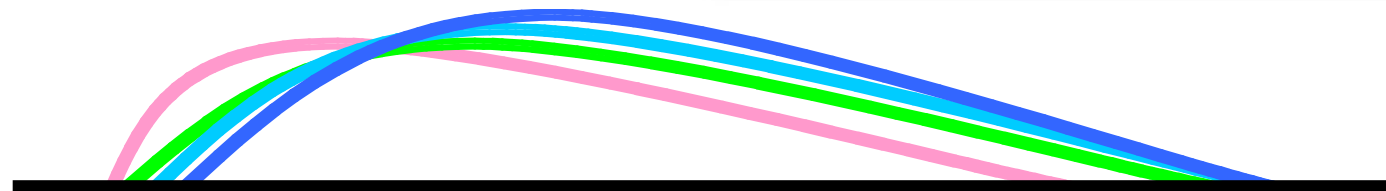


# Sequential vs. overlapping development



Rather than doing all of one thing at a time...

...Scrum teams do a little of everything all the time

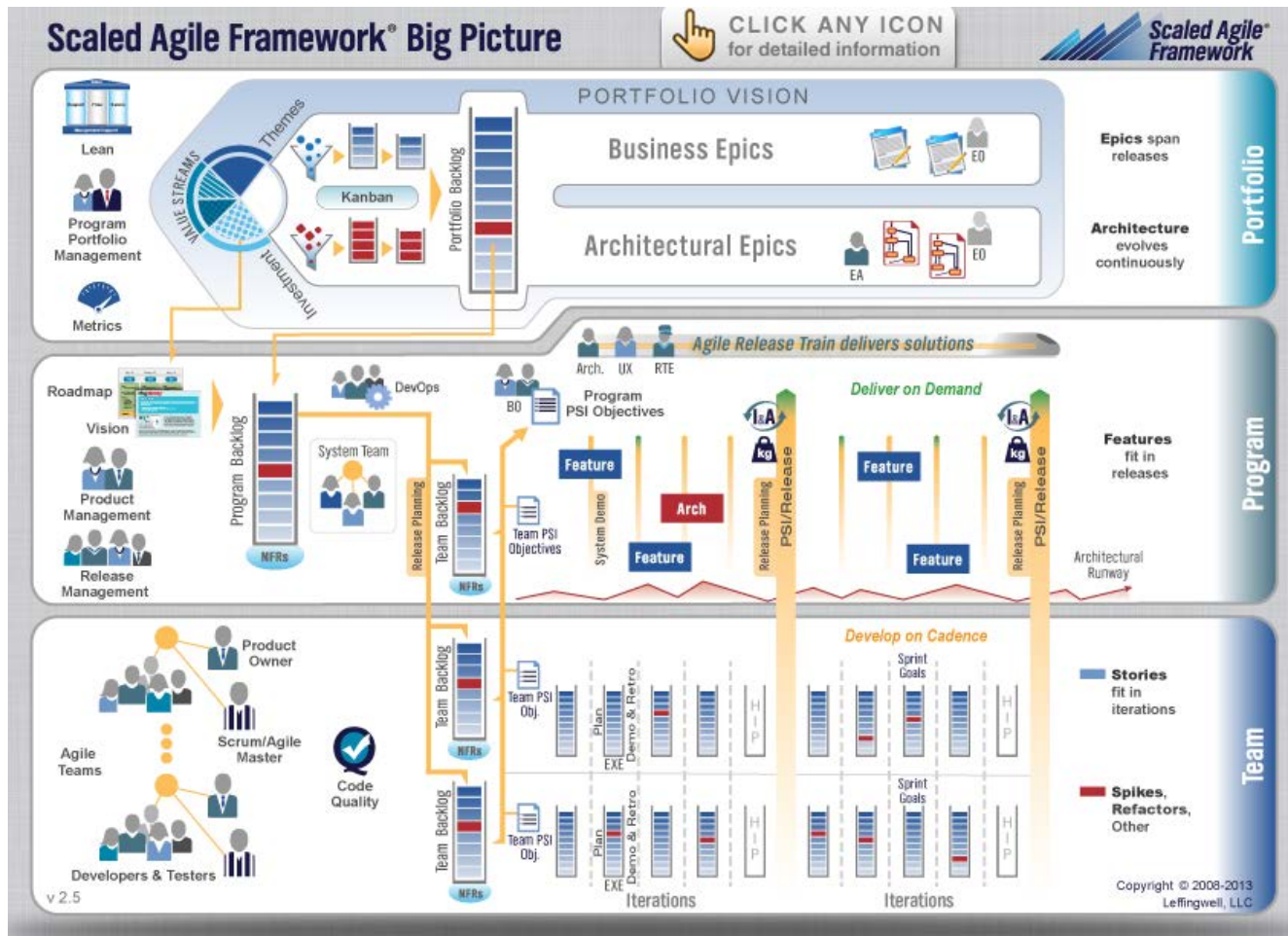


Source: "The New New Product Development Game" by Takeuchi and Nonaka. *Harvard Business Review*, January 1986.





# Agile at Scale-Scaled Agile Framework (SAFe)



Accounts for the strategic business layer, applying lean concepts

Adapts Agile and lean methods to deal with program/project and system issues

Adds some lean concepts to traditional Agile team methods to align team work with larger program

SAFe

[www.scaledagileframework.com](http://www.scaledagileframework.com)



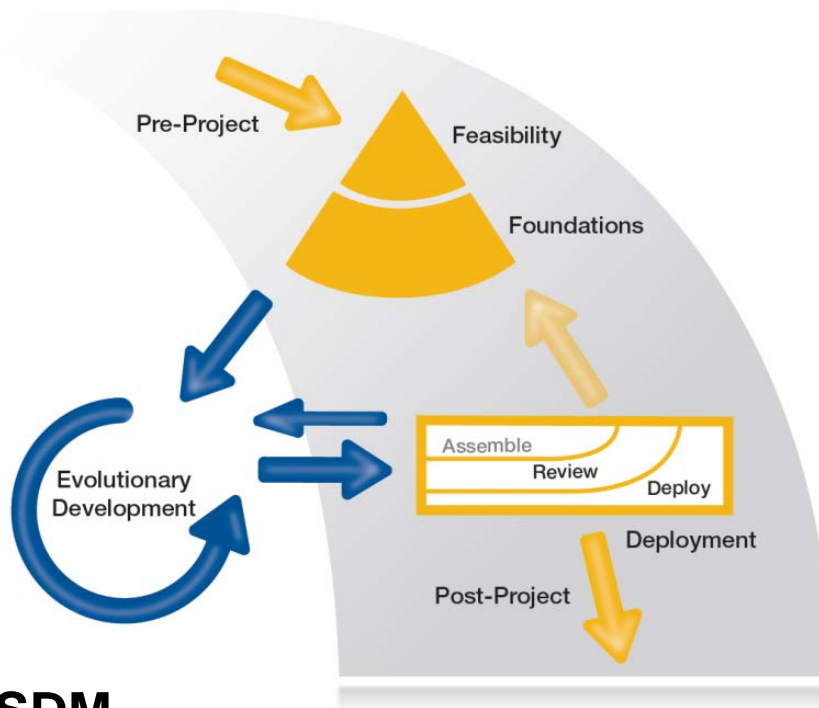
Software Engineering Institute

Carnegie Mellon University

Agile Myths, Picatinny Arsenal  
Lapham, Wrubel Jan 2015  
© 2015 Carnegie Mellon University.

# Agile at Scale-DSDM

## DSDM System Life Cycle



**DSDM**

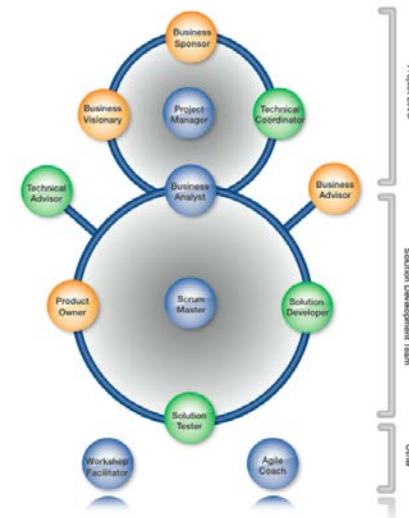
[www.dsdm.org](http://www.dsdm.org)

## 8 Principles for DSDM

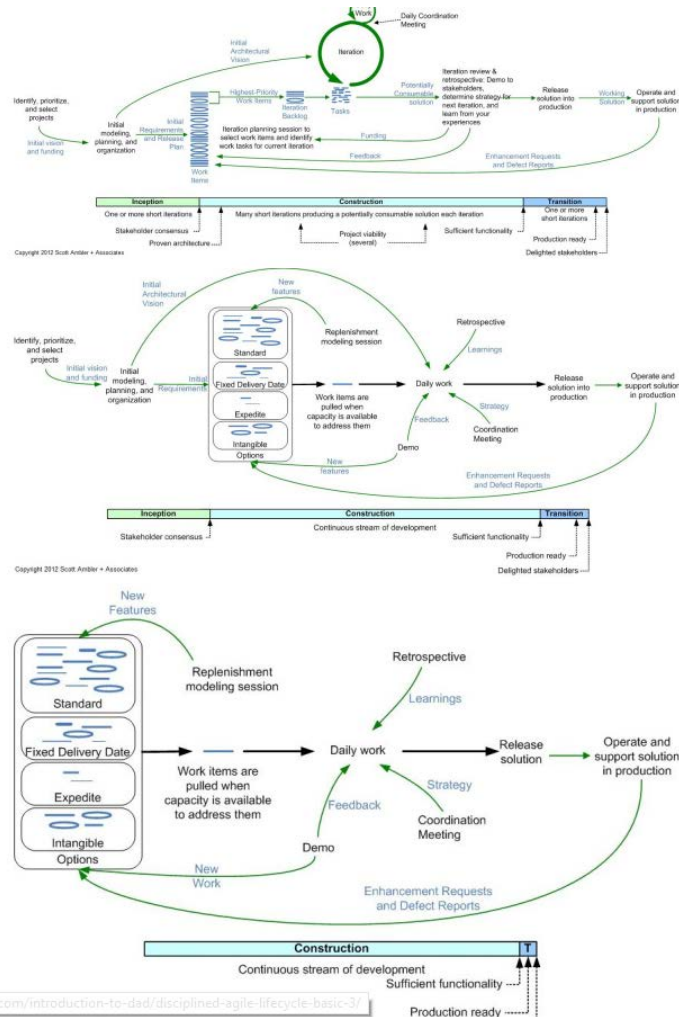
The eight Principles are:

1. Focus on the business need
2. Deliver on time
3. Collaborate
4. Never compromise quality
5. Build incrementally from firm foundations
6. Develop iteratively
7. Communicate continuously and clearly
8. Demonstrate control

## DSDM Roles Map Well to Traditional Project Mgmt



# Agile at Scale—Disciplined Agile Delivery



Leverages concepts from Rational Unified Process and designed to easily align with projects using Rational Unified Process

Risk+value-driven life cycle

Enterprise Aware

Focused at project level



# Extreme Programming – Commonly Used Technical Practices

User stories, used in several Agile methods, derive from Extreme Programming

Technical practices from XP commonly incorporated into other Agile methods:

- Continuous integration
- Daily Build/Automated Regression Test

Pair programming, another XP technical practice, is not used as often as the above outside of XP environments



# Test Driven Development (TDD)

Two common flavors of TDD:

- Within an iteration, at unit/component level
- Acceptance TDD – across the life cycle

## Unit-level TDD

- For each requirement/user story, a test is written BEFORE coding starts on that element
- The minimum amount of code needed to pass the test is written and integrated into the code base

## Acceptance TDD

- Expands the role of the product backlog to include the acceptance tests that will demonstrate satisfaction of the requirements
- Usually user story-based
- Cross-functional teams collaborate to build the acceptance criteria for the stories/requirements



# Crystal

## From A. Cockburn's description of Crystal:

“**Crystal** is a family of human-powered, adaptive, ultralight, “stretch-to-fit” software development methodologies.

- “Human-powered” means that the focus is on achieving project success through enhancing the work of the people involved (other methodologies might be process-centric, or architecture-centric, or tool-centric, but Crystal is people-centric).
- “Ultralight” means that for whatever the project size and priorities, a Crystal-family methodology for the project will work to reduce the paperwork, overhead and bureaucracy to the least that is practical for the parameters of that project.
- “Stretch-to-fit” means that you start with something just smaller than you think you need, and grow it just enough to get it the right size for you (stretching is easier, safer and more efficient than cutting away).

Crystal is non-jealous, meaning that a Crystal methodology permits substitution of similar elements from other methodology

Source: <http://alistair.cockburn.us/Crystal+methodologies>



# Summary

The family of Agile methods has grown in the last 12 years to incorporate methods that address team, project, and enterprise levels of scaling

- It is likely there will never be a “single” Agile method

Some methods are seen more frequently in regulated/government settings than others

All derive from the Agile tenets and principles

Hybrids of multiple methods are common in practice in both industry and government

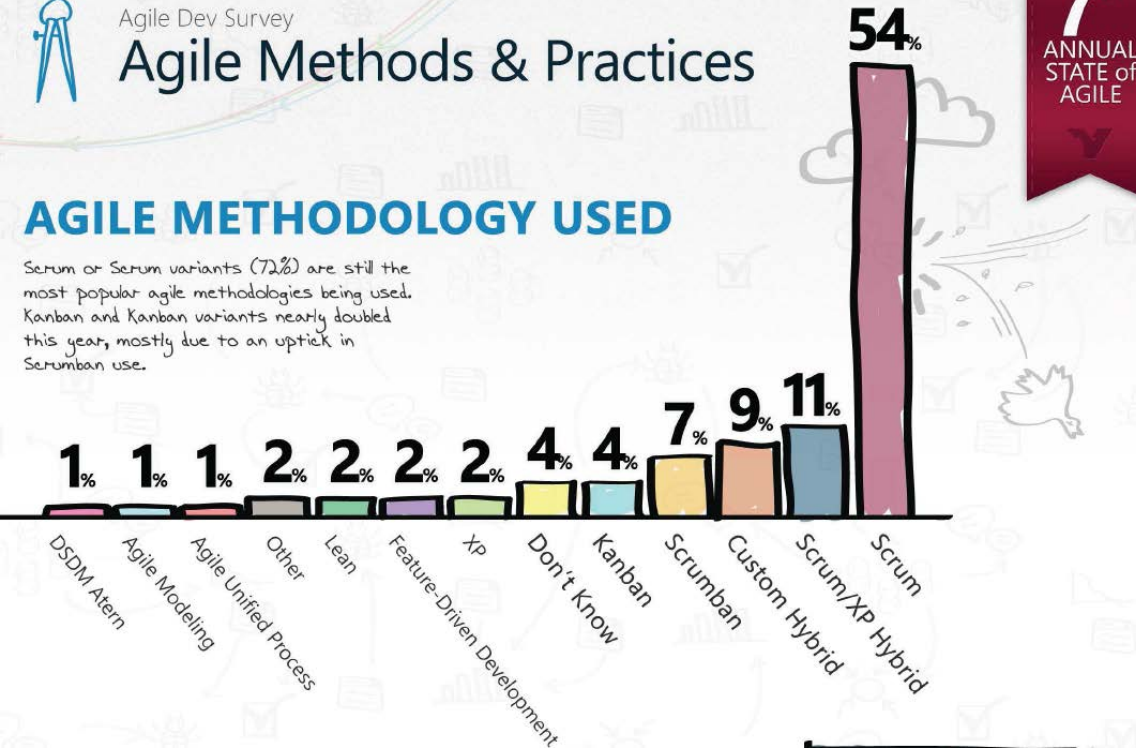






## AGILE METHODOLOGY USED

Scrum or Scrum variants (72%) are still the most popular agile methodologies being used. Kanban and Kanban variants nearly doubled this year, mostly due to an uptick in Scrumban use.



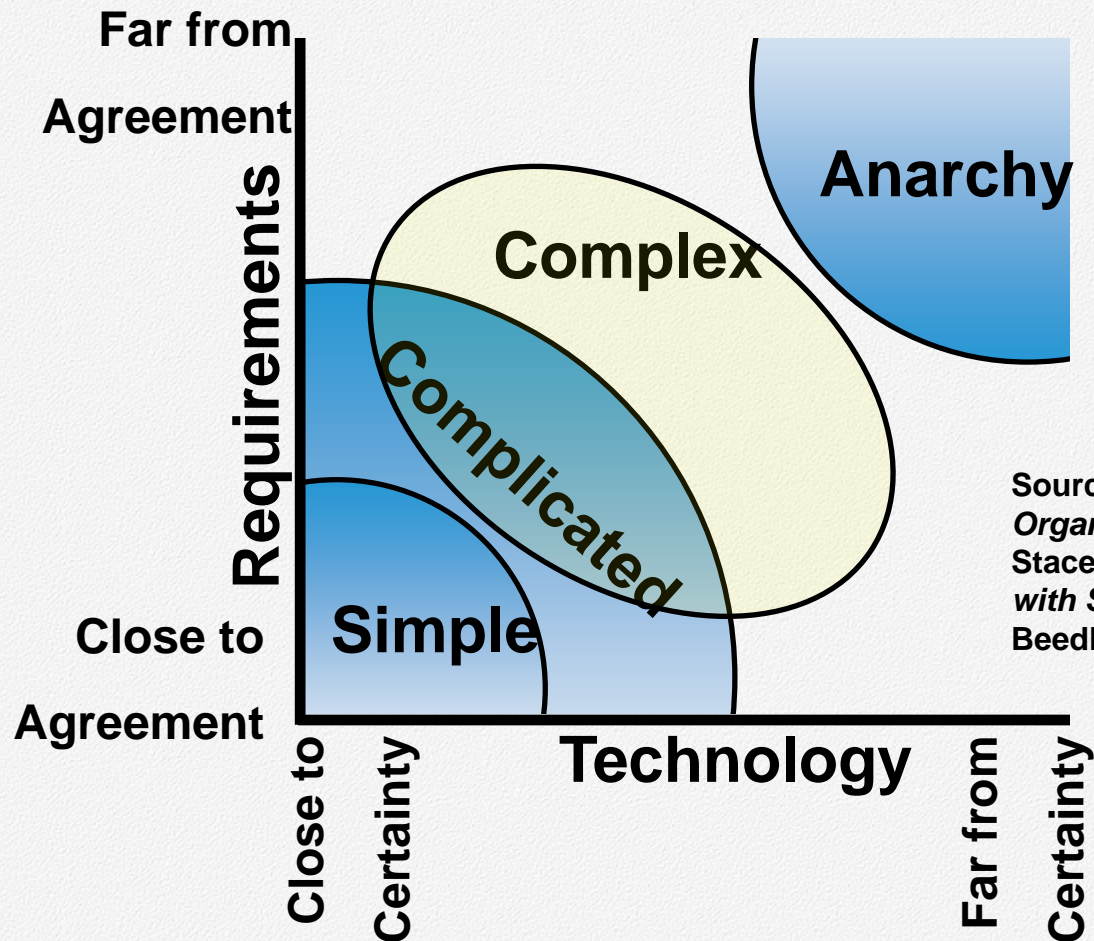
Source: 7<sup>th</sup> Annual Survey on State of Agile, Version One

# SCRUM-THE MOST ADOPTED AGILE METHOD

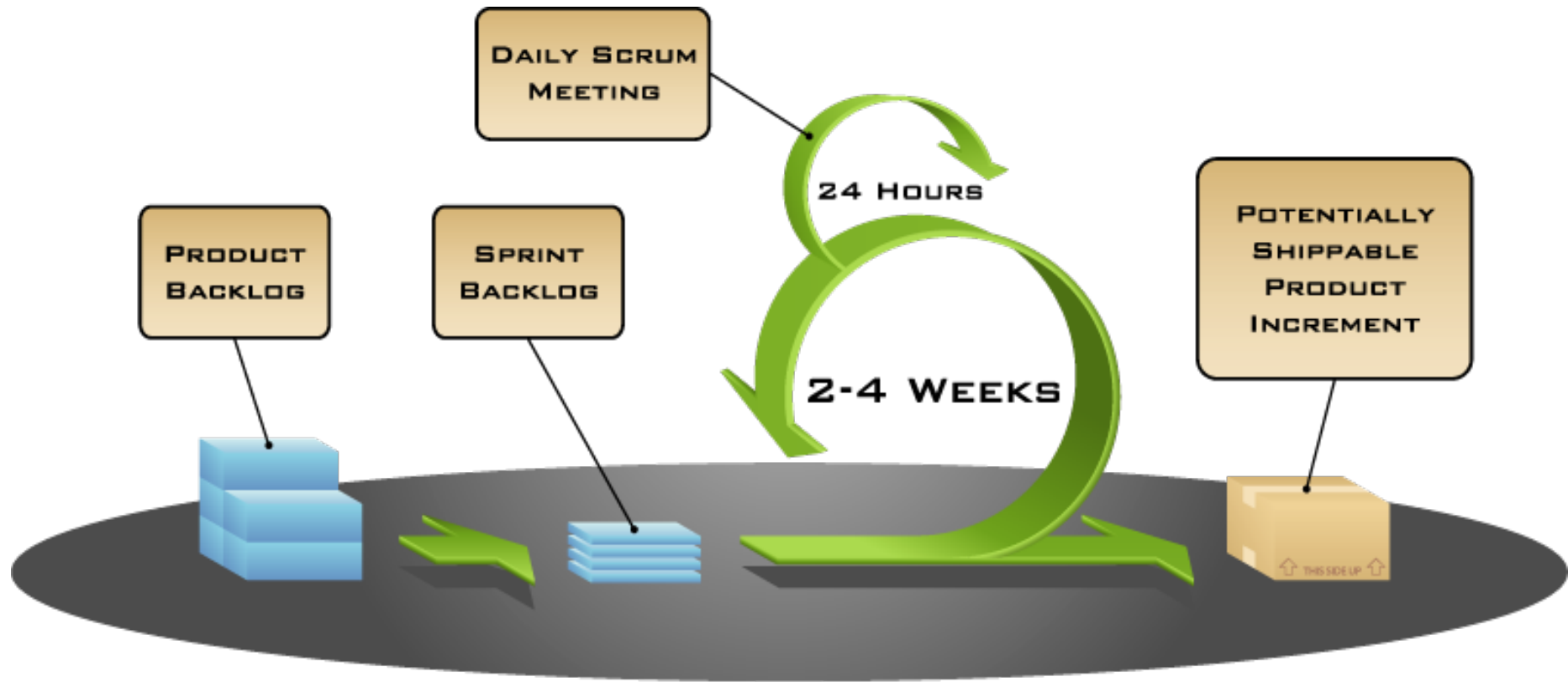




# Sweet Spot for Scrum



# Key Elements of Scrum



COPYRIGHT © 2005, MOUNTAIN GOAT SOFTWARE

Image available at [www.mountaingoatsoftware.com/scrum](http://www.mountaingoatsoftware.com/scrum)



# Scrum framework

## Roles

- Product owner
- ScrumMaster
- Team

## Artifacts

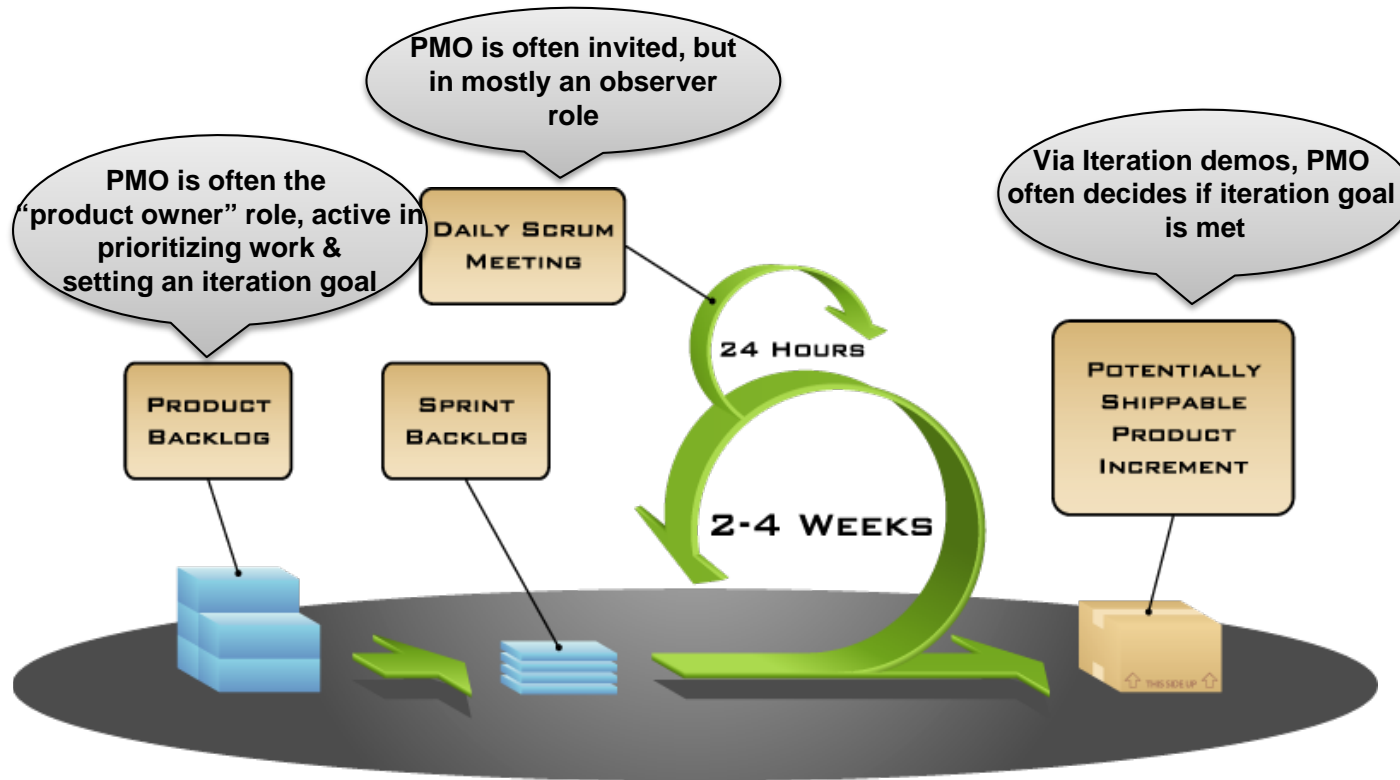
- Product backlog
- Sprint backlog
- Burndown charts

## Ceremonies

- Release Planning
- Sprint planning
- Sprint review
- Sprint retrospective
- Daily scrum meeting



# Where PMO Generally Fits In a Scrum Implementation



COPYRIGHT © 2005, MOUNTAIN GOAT SOFTWARE

*Iterations are often called “sprints”; 2 week sprints are common in industry; 3-4 week sprints are more common in regulated settings*



*How do you decide how long an iteration should be?*



# Summary

Scrum focuses on the team management aspects of Agile software development

As the most commonly-practiced Agile methodology, it is the one that most practitioners are familiar with

Many of the scaling approaches leverage Scrum practices as the team component of their methodology

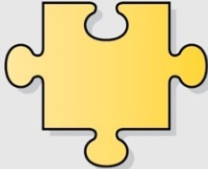
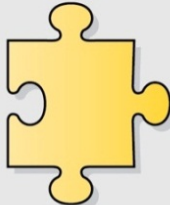
The Scrum “Product Owner” role provides a proactive role for acquisition program offices to collaborate with software teams



# CHALLENGES TO AGILE ADOPTION IN DOD



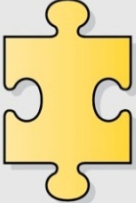

# Comparison of Agile and Traditional DoD Cultural Elements<sub>1</sub>

Knowledge Piece	Method	
<b>Organizational Structure</b> 	<b>Agile DoD</b> <ul style="list-style-type: none"><li>• Flexible and adaptive structures</li><li>• Self-organizing teams</li><li>• Collocated teams or strong communication mechanisms when teams are distributed</li></ul>	<b>Traditional DoD</b> <ul style="list-style-type: none"><li>• Formal structures that are difficult to change</li><li>• Hierarchical, command-and-control-based teams</li><li>• Integrated product teams that have formal responsibilities</li></ul>
<b>Leadership Style</b> 	<b>Agile DoD</b> <ul style="list-style-type: none"><li>• Facilitative leadership</li><li>• Leader as champion and team advocate</li></ul>	<b>Traditional DoD</b> <ul style="list-style-type: none"><li>• Leader as keeper of vision</li><li>• Leader as primary source of authority to act</li></ul>

<http://www.sei.cmu.edu/library/abstracts/reports/11tn002.cfm?DCSext.abstractsource=SearchResults>



# Comparison of Agile and Traditional DoD Cultural Elements<sub>2</sub>


Knowledge Piece	Method	
<b>Rewards System</b> 	<b>Agile DoD</b> <ul style="list-style-type: none"> <li>• Team is focus of reward systems</li> <li>• Sometimes team itself recognizes individuals</li> </ul>	<b>Traditional DoD</b> <ul style="list-style-type: none"> <li>• Individual is focus of the reward system</li> </ul>
<b>Staffing Model</b> 	<b>Agile DoD</b> <ul style="list-style-type: none"> <li>• Cross-functional teams including all roles across the life cycle throughout the lifespan of the project</li> <li>• Includes an Agile advocate or coach who explicitly attends to the team's process</li> </ul>	<b>Traditional DoD</b> <ul style="list-style-type: none"> <li>• Uses traditional life-cycle model with separate teams, particularly for development and testing</li> <li>• Different roles are active at different defined points in the life cycle and are not substantively involved except at those times</li> </ul>

<http://www.sei.cmu.edu/library/abstracts/reports/11tn002.cfm?DCSext.abstractsource=SearchResults>





# Comparison of Agile and Traditional DoD Cultural Elements<sub>3</sub>

Knowledge Piece	Method	
<p><b>Communications &amp; Decision Making</b></p> 	<p><b>Agile DoD</b></p> <ul style="list-style-type: none"> <li>• Daily stand-up meetings</li> <li>• Frequent retrospectives to improve practices</li> <li>• Information radiators to communicate critical project information</li> <li>• Evocative documents to feed conversation</li> <li>• “Just enough” documentation, highly dependent on product context</li> </ul>	<p><b>Traditional DoD</b></p> <ul style="list-style-type: none"> <li>• Top-down communication structures dominate</li> <li>• External regulations, policies and procedures drive the focus of work</li> <li>• Indirect communications, like documented activities and processes, dominate over face-to-face dialogue</li> <li>• Traditional, representational documents used by the PMO throughout the development life cycle to oversee the progress of the developer</li> <li>• PMO oversight tools focused on demonstrating compliance vs. achieving insight into progress</li> </ul>

<http://www.sei.cmu.edu/library/abstracts/reports/11tn002.cfm?DCSext.abstractsource=SearchResults>



# Exercise



Break into pairs and discuss which of the “Agile DoD” cultural elements you think are the most difficult for your DoD setting.

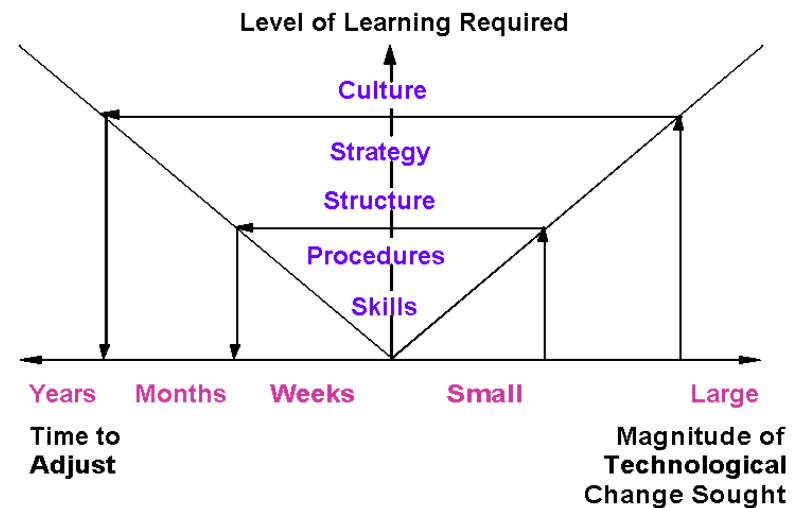
Be prepared to discuss your thoughts with the larger class.



# Polling Question

## How Big a Challenge is Your Adoption of Agile Practices?

- **Large**, we need a culture change
- **Medium**, we are running into issues
- **Small**, we are mostly ready
- **Nonexistent**, no challenge at all

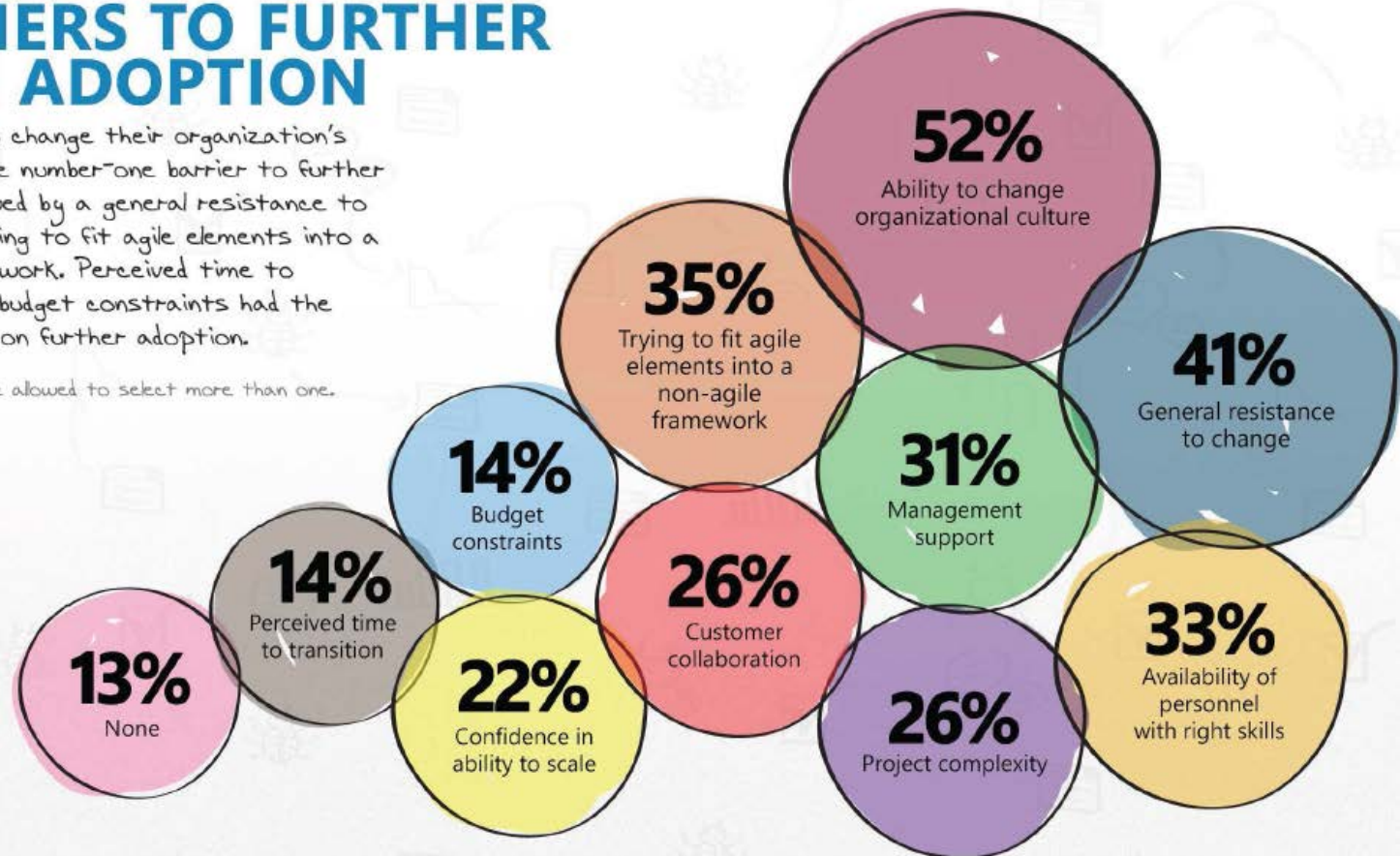


# Culture Change is an Issue throughout Industry

## BARRIERS TO FURTHER AGILE ADOPTION

The inability to change their organization's culture was the number-one barrier to further adoption, followed by a general resistance to change and trying to fit agile elements into a non-agile framework. Perceived time to transition and budget constraints had the lowest impact on further adoption.

\*Respondents were allowed to select more than one.



# Agile Software Development

Reasons for Failure	Paths to Success
Inadequate oversight	Working <i>shoulder to shoulder</i> for PMOs and developers
Inadequate requirements management	<b>Continuous</b> involvement of users and other stakeholders, <i>paying attention to end-to-end capability</i>
Inadequate design and architecture	<b>Integrated</b> design, implementation and integration cycles on top of an <i>infrastructure enabling agility</i>
Inadequate documentation	<b>Consolidated documentation</b> forms that <i>minimize redundancy</i> Artifacts that <b>enhance communication</b> and maintainability





# Patriot Excalibur: An Agile Success Story in DoD

## The Agile Journey of Patriot Excalibur (PEX)

A case of "Fortunate Serendipity"

- Reimbursable funding model
- Beneath the acquisition radar in the test wing



### Adopted XP in 2008

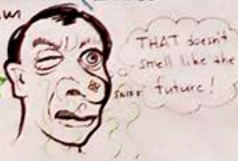
- o more SMEs
- o more releases
- o more functionality delivered

Went from  
20 to 200  
adapters in  
ONE YEAR!



### 2007 Agile Struggles

- o More demand led to a bigger team
- o 20 on a team was too big!
- o "traditional" smells crept in



### 2008 Agile ReOrg 1

- o Quasi-Scrum model
- o Independent teams
- o More SMEs
- o Add in SCM

- ★ MISSION REASSIGNED TO AFLCMC "Agile Evangelists"
- o Accredited as a "software program" (like MS Word) instead of a system

### 2009/10 Expansion

- o Demand increased to 600 adapters
- o Increased # of teams ~ 100 people
- o Architecture → SOA
- o Geographically distributed team
- o Added automated testing team

### 2010 Difficulty Scaling

- o 7 week test event @ end of 8 iterations
- o teams still very independent
- o introduced technical debt that built up

### 2011 Agile ReOrg 2

- o Adopted Scrum "by-the-book"
- o Product Owner Team
- o Embedded Tester
- o More automation
- o Integrated Design Teams
- o Stabilized architecture @ end of iteration

DEFINITION OF DONE



### 2012-2013 Scrum

- o Host @ 16 locations
- o Every MATCOM uses it
- o Moved to GSA T&M contract - SW dev as a service
- o Single code base still a key
- o Added specialized skill of security
- o Multiple technologies side-by-side: Web & Client Server ~ 2 million SLOC
- o AFLCMC Support

### Open Issues

- o Architecture Committee is inefficient
- o Long-range planning
- o 7-week system test at the end
- o Reliable funding stream
- o Code quality & Test Coverage

©2013 Software Engineering Institute

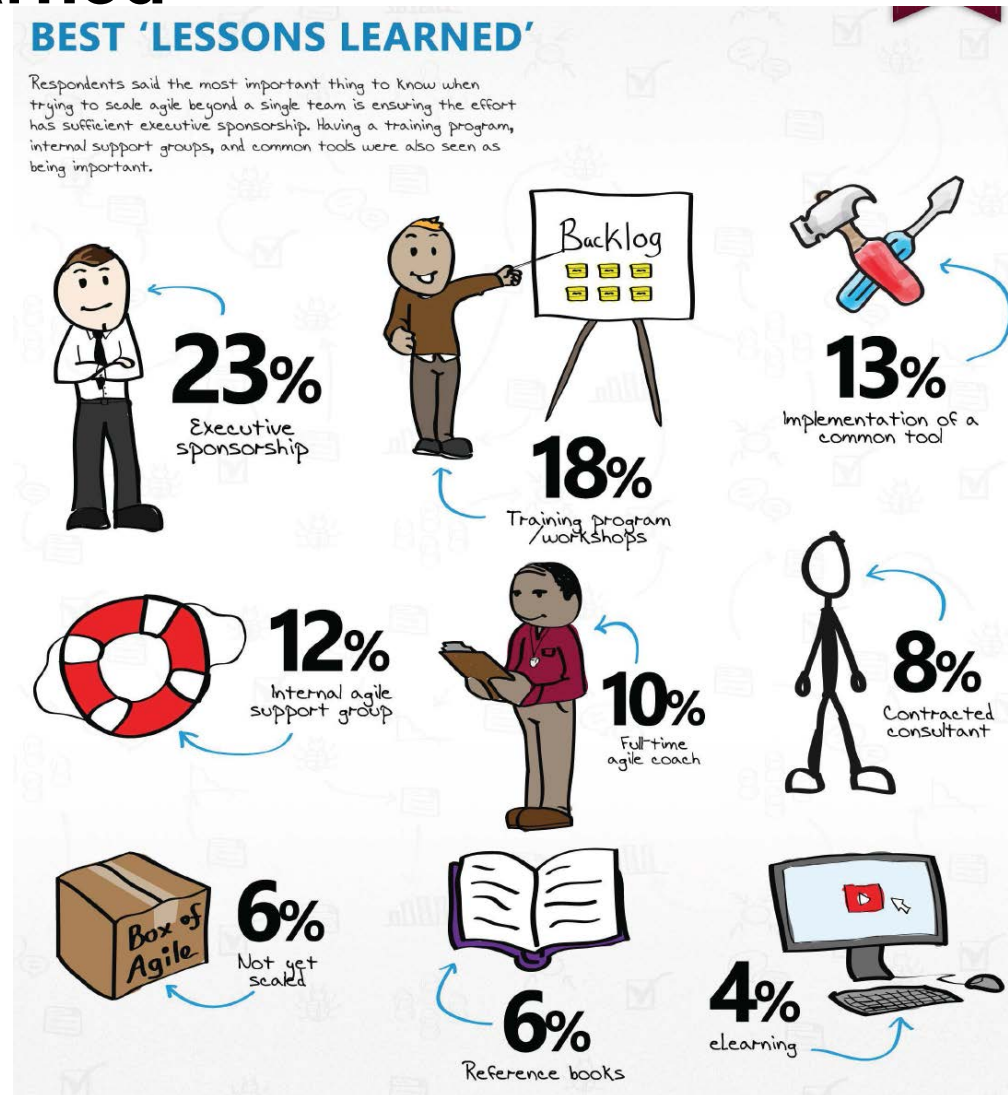


# **SUGGESTIONS FOR SUCCESSFUL USE OF AGILE METHODS IN DOD ACQUISITION**



# Industry Lessons Learned

Many of the lessons learned in commercial industry for Agile adoption can apply in DoD settings



Source: 7<sup>th</sup> Annual Survey on State of Agile, Version One





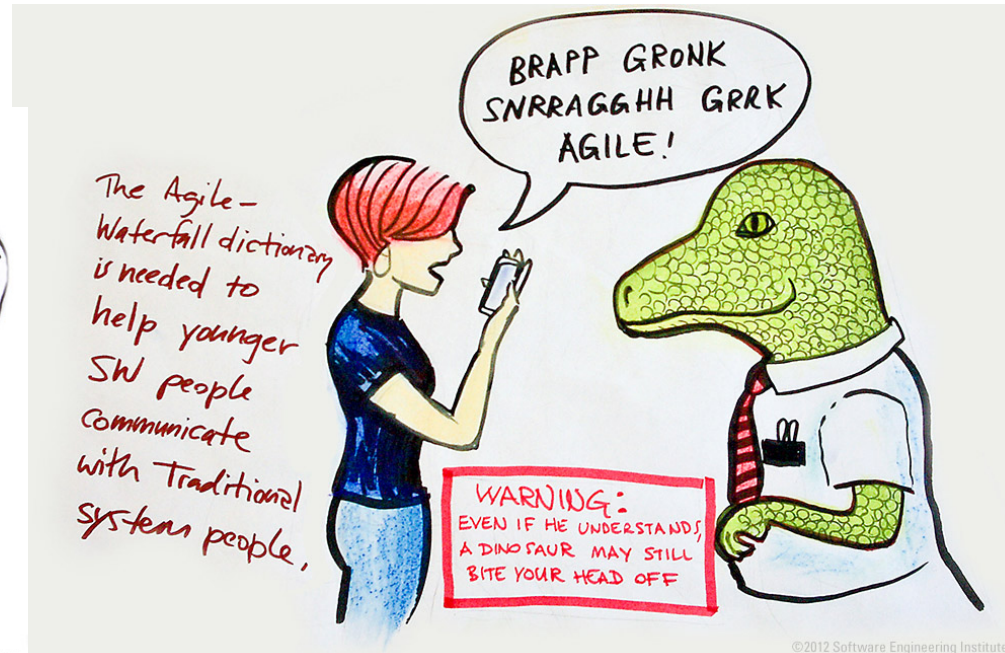
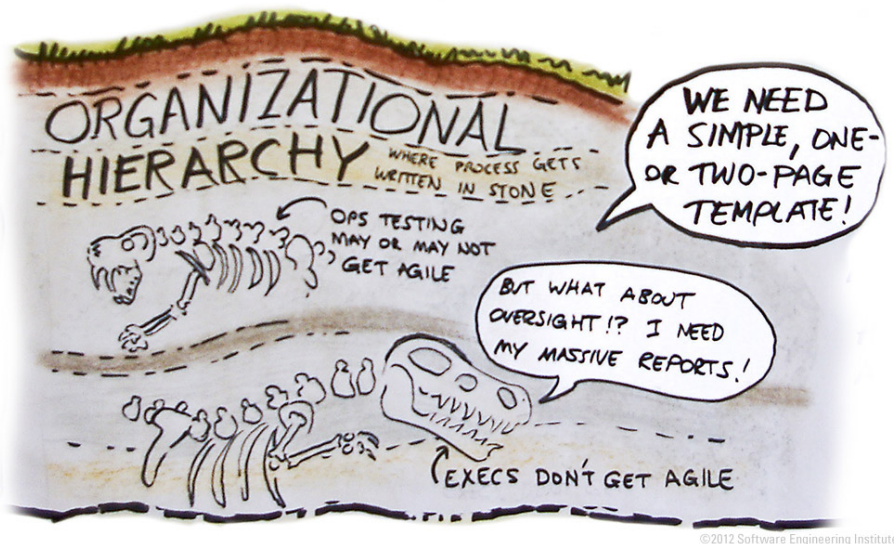
# Suggesting Successful Approaches

Educating leadership and staff on differences they will see

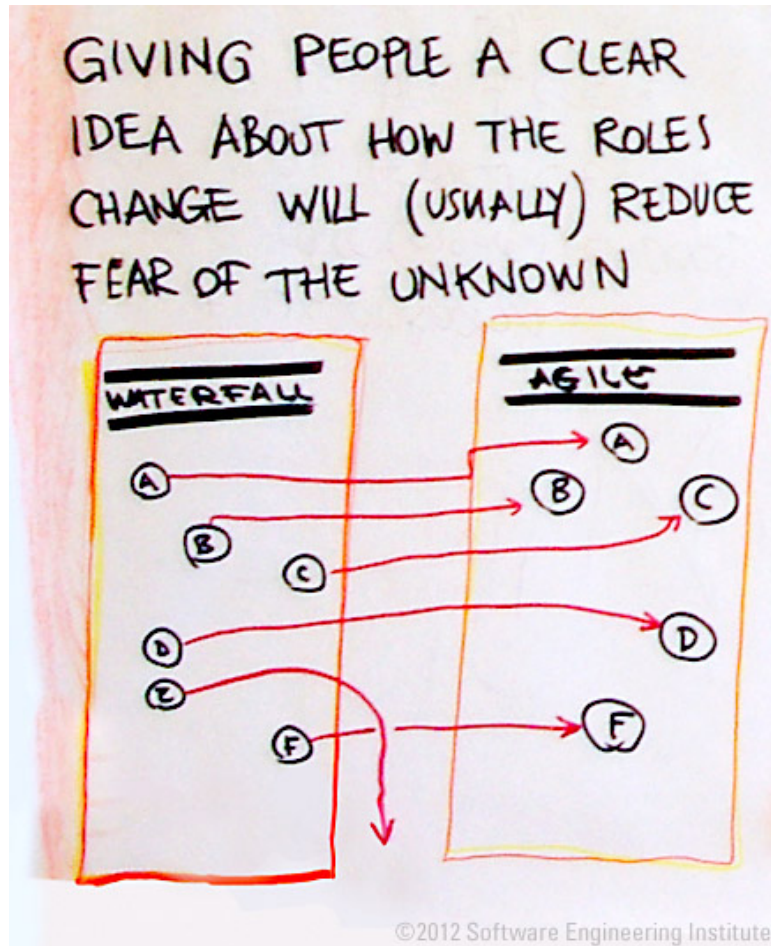
Reminding organizations of the typical challenges they face for a big change

Disseminating successful approaches when we find them

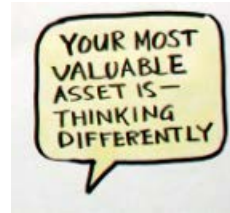
Adding in a little humor along the way...



# Help People Understand What's Coming



# A Few Things to Think About When People Cite “Regulations” as a Reason *\*Not\** to Embrace Agile Approaches



Who are the real stakeholders? Where are the political “bodies” buried? How do the “ghosts” in the stakeholder map affect what people do today?

Value stream mapping, a lean technique, is sometimes useful to point out waste areas where Agile could help, in an organization that is trying to reduce cost by eliminating wasted effort

When analyzing processes currently in use, always ask “For whom?” and “So what?” (ask them about your Agile practices too!!)

Most regulated environments involve conflicting mandates: different people choose different areas to emphasize—try to find the ones most complementary to Agile approaches and focus on those





# Caution: Burning Platform Approach to New Practice Adoption

It can work, but there are lots of challenges.



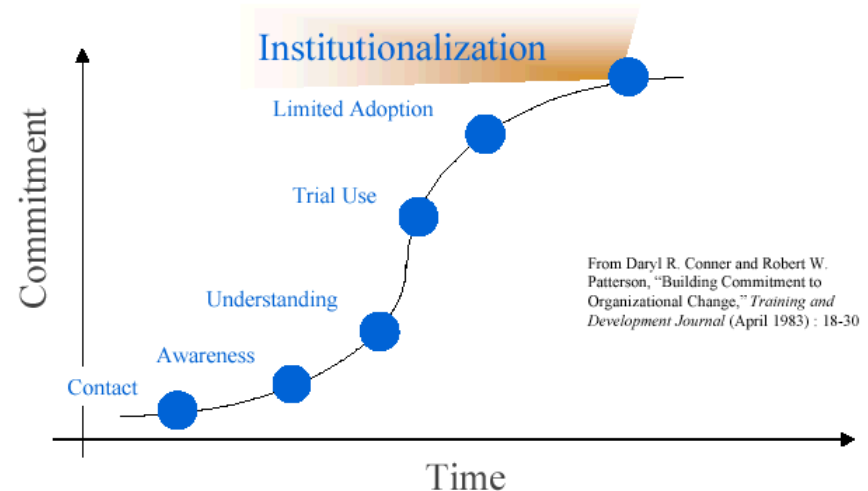
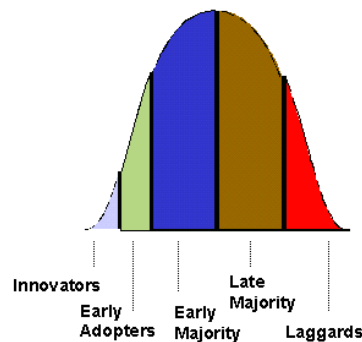
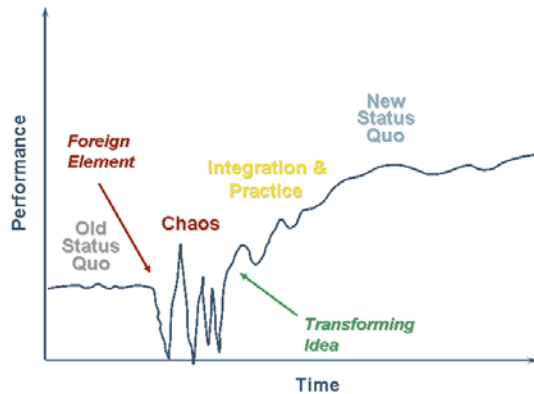
©2013 Software Engineering Institute



# “Traditional” Adoption Tools and Methods Work Well with Agile Adoption

Understand the Change Cycle and Your Adoption Population

Prepare for Both Communication and Implementation Support Mechanisms that are Needed



# Exercise



Break into pairs and discuss communication and implementation mechanisms you think will be necessary to adopt (or were missing, if you've already adopted) Agile methods in your setting. Mark them with the initial of which stage of the Adoption Commitment Curve you believe they address.

Be prepared to discuss your thoughts with the larger class.

## Communication Mechanisms

- C=Contact/Awareness
- U=Understanding

## Implementation Mechanisms

- TU=Trial Use
- A=Limited Adoption
- I=Institutionalization

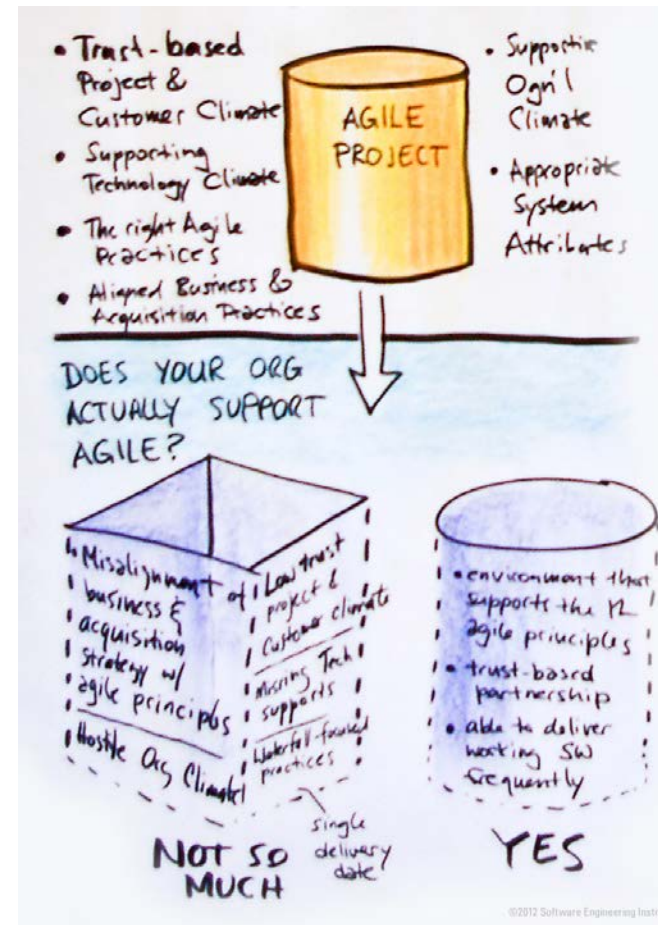


# Understanding Your Readiness for Agile Adoption

General cultural analyses for adoption of Agile methods don't tend to pick up some of the acquisition issues inherent in these environments.

SEI Readiness & Fit Analysis (RFA) and its underlying model explicitly include risk areas known to impede Agile adoption in regulated environments

- More emphasis on business models, goal alignment, and acquisition strategy
- More focus on alignment issues—especially related to staff turnover
- Some particular issues around interfacing with systems engineering in large systems developments



# Agile RFA Categories

- Business and acquisition** —adoption factors related to business strategy, acquisition strategy, and contracting mechanisms

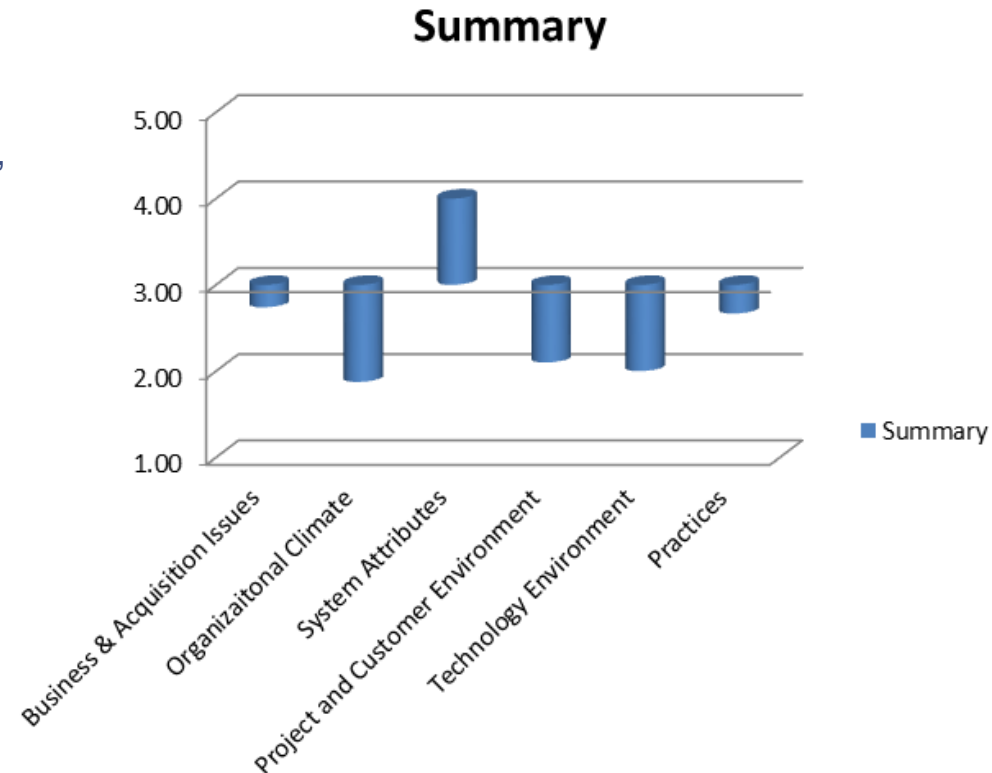
- Organizational climate** – adoption factors related to sponsorship, leadership, reward systems, values, and similar “soft” issues

- System attributes** – adoption factors related to the actual characteristics of the system(s) being developed

- Project and customer environment** – adoption factors related to project management norms, team dynamics and support structures, and customer relationships and expectations

- Technology environment** – adoption factors related to the technologies that are in place or planned to support the selected Agile methods

- Practices** – a taxonomy of Agile practices commonly adopted within DoD that is used to understand which practices an organization plans to adopt so that other factors can be calibrated around those expectations



*Rating of 3 indicates some issues likely, but nothing unusual for an Agile adoption. Below 3 indicates issues that will negatively impact adoption success. Above 3 indicates issues that could enable adoption success.*

*More important than the rating is the specific risks that RFA participants identify.*

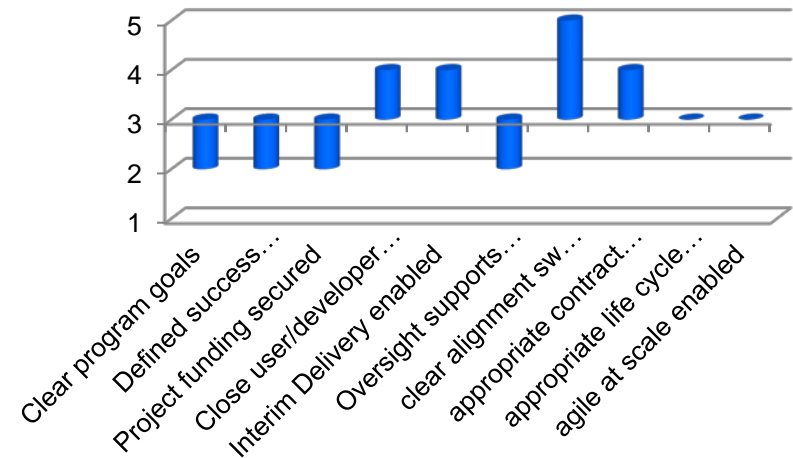




# Business & Acquisition Category Particularly Affects DoD Agile Adoption

- Business or Program goals are clear and reflect stakeholders concerns
- Success strategies (e.g. roadmaps, product portfolios) are defined and clearly communicated
- Funding for the project has been secured
- Mechanisms are in place in the contract and acquisition strategy to allow close collaboration between developers and end users
- Mechanisms are in place in the contract and acquisition strategy that allow for interim demonstration and delivery between official releases
- Contract oversight mechanisms are aligned with agile principles
- The alignment of software-related goals with program-level goals is clear
- Contract type accounts for use of agile/lean methods in the program
- Life cycle activities are planned in the acquisition strategy that are compatible with agile/lean methods
- Acquisition strategy takes into account the use of agile methods at the scale needed for the program

**Business/Acquisition Factors**



# SEI's Agile RFA Application So Far

Agile RFA has been applied by the SEI in 2 DoD and 1 Federal Agency setting

- Case 1 was a military program considering adopting Agile methods
  - They identified areas to address to improve their chance of success
- Case 2 was a military program already adopting Agile methods
  - They identified the source of symptoms they were aware of
  - They also identified disconnects between management and practitioner views of what was going on
  - They used RFA to help them monitor adoption progress and systematically reduce probability and impact of adoption risks
- Case 3 was a federal agency who was mandated to adopt Agile methods
  - They identified areas to work on before starting Agile pilots
  - They used RFA to help them monitor adoption progress



# Summary

Adopting Agile methods involves (sometimes significant) cultural shifts as well as practice changes

- Transition and adoption approaches for other major organizational changes will work for Agile adoption as well
- Go in with your eyes open—Readiness & Fit Analysis or other methods of assessing readiness can help you to avoid pitfalls as you approach adoption
- Many adoption support mechanisms exist out in the commercial world that can be adapted to regulated settings
  - There are even training courses geared toward government settings becoming available
  - The SEI Technical Notes and other resources (blogs, podcasts, etc) on Agile adoption are meant to support acquisition practitioners in becoming knowledgeable about different issues they may encounter when adopting Agile or lean methods



# SUMMARY



# Agile Myths—BUSTED!



Remember? Jamie and I testing Killer Washing Machine. One of the few where NOTHING was true. -- Adam (@DontTryThis)

Image Credit: DCI



# Remember...Agile is Not a Silver Bullet



# Call to Action

If the things we have discussed resonate with you

- Read some of our papers/presentations to get ideas
  - Every Technical Note represents inputs from dozens of people actively working in adopting Agile methods in a DoD or federal setting
- See where strategies that have worked in other parts of the DoD might apply to you
  - Look for places from our work where you have faced similar situations
  - If you have a success strategy you don't see us promulgating, please consider sharing with us! We're here to learn!
- Consider joining our Agile Collaboration Group
  - Over 125 members from all military services, contractors, federal agencies who help us focus our research and share their experiences
    - Send email to Mary Ann Lapham, [mlapham@sei.cmu.edu](mailto:mlapham@sei.cmu.edu)



# Thank you!!!





# Contact Information

**Mary Ann Lapham**

Principal Engineer

Client Technical Solutions

Email: [mlapham@sei.cmu.edu](mailto:mlapham@sei.cmu.edu)

**Eileen Wrubel**

Client Technical Solutions

Email: [eow@sei.cmu.edu](mailto:eow@sei.cmu.edu)

## U.S. Mail

Software Engineering Institute

Customer Relations

4500 Fifth Avenue

Pittsburgh, PA 15213-2612

USA

## Web

[www.sei.cmu.edu/acquisition/research](http://www.sei.cmu.edu/acquisition/research)

## Customer Relations

Email: [info@sei.cmu.edu](mailto:info@sei.cmu.edu)

SEI Phone: +1 412-268-5800

SEI Fax: +1 412-268-6257



# BACKUP SLIDES



# **BONUS:** **GIMMES AND GOTCHAS**



# Gimmes & Gotchas

***Gimmes*** are a list of behaviors that give you confidence that your program office, organic development organization, and/or contractor is embracing an agile process.



***Gotchas*** are a list of behaviors that may indicate problems currently exist or are on the horizon in your agile program.

These Gimmes & Gotchas **are not** intended to be all inclusive, nor are they a checklist. The goal of these is to help identify areas to investigate further and focus your energy toward a successful program.



# First Steps Gimmes <sub>1</sub>



- Your program office, organic development teams, and contractors understand the system is software-reliant
- *Your motivation, trade-offs, benefits, and expectations for using an agile approach is clearly documented and understood*
- There is an *explicit* understanding that the requirements are expected to evolve
- Automated testing is planned for and budgeted
- Multiple acquisition approaches are being considered

*Italics – this item also impacts external stakeholders*



# First Steps Gimmes <sub>2</sub>



- Contract/MOA allows flexibility and incremental delivery
- The concept of incremental delivery of content is included in CDRLs or MOA terms
- Entire program team is aware that the DoD 5000.02 has two new lifecycle descriptions that support more "agile" approaches
- Hindrances for agile implementation are acknowledged and paths to success are identified
- Readiness for agile adoption is determined by using a formal method such as the SEI Readiness and Fit Analysis (RFA)





# First Steps Gotchas

- *Senior managers and stakeholders reluctantly agreed to use agile or are unaware*
- Software development is constrained to a hardware architecture
- *Mindset that document completion equals progress*
- Program exists in a "risk averse culture"
- Integration testing isn't planned until just before final delivery
- Testing isn't budgeted until much later in the program
- *Agile is being considered a silver bullet*

*Italics – this item also impacts external stakeholders*



# Readiness Gimmes <sub>1</sub>



- Your agile approach has been tailored to best meet your program's needs
- Program office staff and organic development team, including systems engineers, understand the agile process you're using
- The agile manifesto and principles are understood throughout the organization
- Appropriate training has been provided for the entire organization
- *Expectations and artifacts necessary for milestone decisions have been agreed to and documented*

*Italics – this item also impacts external stakeholders*





# Readiness Gimmes <sub>2</sub>



- Agile roles and responsibilities have been clearly assigned
- *The definition of done has been established and includes what documentation is required*
- The contracting team has been trained and understands the agile process
- The program office is open to changing roles
- Systems engineers are an integral part of the agile process

*Italics – this item also impacts external stakeholders*



# Readiness Gimmes <sub>3</sub>



- The schedule identifies when emulators/simulators are needed by the software development team
- Leadership and staff are educated on differences from the way they are used to doing business
- Program team/organic development team has answers for most agile "myths"
- Program/organic development team utilizes adoption support mechanisms, e.g. coaching/training, facility configuration that supports information radiators, industry blogs/publications, etc.





# Readiness Gotchas

- Your testing function/organization has not been integrated into the day-to-day activities
- Requirements stability, operating environment, and the evolution of the technology base has not been fully assessed
- *Constraints are imposed for the sake of tradition*
- *Contract progress payments are based on "earned value" for the accounting period*
- Testing organization is not involved in the agile process
- Regulations are cited as a reason not to embrace agile approaches

*Italics – this item also impacts external stakeholders*



# Application Gimmes <sub>1</sub>



- *Your users and stakeholders can accommodate incremental deliveries*
- *Necessary and beneficial documentation has been identified*
- *Requirements can be prioritized without pushback*
- User stories conform to the “INVEST”\* concept
- *Technical reviews are structured to understand technical issues and mitigate technical risks*

\* INVEST = Independent, Negotiable, Valuable, Estimable, Small, Testable

*Italics – this item also impacts external stakeholders*



# Application Gimmes <sub>2</sub>



- *Iteration and release reviews are used to build a case to demonstrate readiness to pass milestone reviews*
- Agile measurements are integrated into your overall management metrics
- Measurements are focused on “are we producing sufficient value fast enough?”
- *User requirements are validated during the creation of user stories*
- The program office responsibilities haven't changed but how they perform them has

*Italics – this item also impacts external stakeholders*





# Application Gotchas

- Your program or contractor is proposing agile as a quick fix for existing failures on the program
- Team metrics (e.g., velocity) are used for comparisons\*
- *Users and stakeholders are not actively engaged in the agile process*
- Oversight activities are abandoned
- Focus is on compliance rather than mission success

\*Team composition varies, so relative measures like velocity apply only *within* a team, not *between* multiple teams.

*Italics – this item also impacts external stakeholders*



# PUBLICATIONS



# Selected SEI Agile Publications<sub>1</sub>

## Papers

**Agile Methods in Air Force Sustainment: Status and Outlook (CMU/SEI-2014-TN-009)**

<http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=312754>

**Considerations for Using Agile in DoD Acquisition (CMU/SEI 2010-TN-002)**

[http://resources.sei.cmu.edu/asset\\_files/TechnicalNote/2014\\_004\\_001\\_312766.pdf](http://resources.sei.cmu.edu/asset_files/TechnicalNote/2014_004_001_312766.pdf)

**Agile Methods: Selected DoD Management and Acquisition Concerns (CMU/SEI-2011-TN-002)**

<http://www.sei.cmu.edu/library/abstracts/reports/11tn002.cfm?DCSext.abstractsource=SearchResults>

**Agile Metrics: Progress Monitoring of Agile Contractors (CMU/SEI-2013-TN-029)**

<http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=77747>

**A Closer Look at 804: A Summary of Considerations for DoD Program Managers (CMU/SEI-2011-SR-015)**

<http://www.sei.cmu.edu/library/abstracts/reports/11sr015.cfm?DCSext.abstractsource=SearchResults>

**DoD Information Assurance and Agile: Challenges and Recommendations Gathered Through Interviews with Agile Program Managers and DoD Accreditation Reviewers (CMU/SEI 2012-TN-024)**

<http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=34083>

**Agile Software Teams: How They Engage with Systems Engineering on DoD Acquisition Programs (CMU/SEI-2014-TN-013)**

<http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=295943>





# Selected SEI Agile Publications<sub>2</sub>

## Papers, cont.

**Parallel Worlds: Agile and Waterfall Differences and Similarities (CMU/SEI-2013-TN-021)**

<http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=62901>

**Potential Use of Agile Methods in Selected DoD Acquisitions: Requirements Development and Management (CMU/SEI-2013-TN-006)**

<http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=89158>

**DoD Agile Adoption: Necessary Considerations, Concerns, and Changes**

<http://www.crosstalkonline.org/issues/janfeb-2012.html>

## Colloquium

SEI Agile Colloquiums, June 20-21, 2012; March 19-20, 2013, July 16-17, 2013, June 5, 2014

Colloquium Graphic Recording (available upon request)

## Webinar

Agile Research Forum, Agile Methods: Tools, Techniques, and Practices for the DoD Community, Mary Ann Lapham, August 2012

<http://www.sei.cmu.edu/go/agile-research-forum/>



# Selected SEI Agile Publications<sub>3</sub>

## **Blogs**

Readiness and Fit Analysis (October 8 2012)

<http://blog.sei.cmu.edu/archives.cfm/author/suzanne-miller>

Agile Methods: Tools, Techniques, and Practices for the DoD Community

<http://blog.sei.cmu.edu/post.cfm/agile-methods-tools-techniques-and-practices-for-the-dod-community>

Using Agile Effectively in DoD Environments (February 6, 2012)

<http://blog.sei.cmu.edu/archives.cfm/author/mary-ann-lapham>

## **Podcast**

Agile Acquisition (September 4, 2012)

<http://www.sei.cmu.edu/podcasts/index.cfm?getRecord=7D03CB1F-9D60-C314-66526F8E8B2864B8&wtPodcast=AgileAcquisition>

DoD and Agile Principles Series

<http://www.sei.cmu.edu/podcasts/agile-in-the-dod/index.cfm>

## **Want More?**

For additional SEI papers, blogs, and podcasts on Agile, please visit the SEI Digital Library at

[http://resources.sei.cmu.edu/library/results.cfm?as\\_q=inmeta:gsataxonomyoutput~Agile](http://resources.sei.cmu.edu/library/results.cfm?as_q=inmeta:gsataxonomyoutput~Agile)

**We regularly publish new materials! Check back often!**



# Selected SEI Agile Publications<sub>4</sub>

## Selected Conferences

- NDIA Systems Engineering Conference October 25-28, 2010
  - Presentation: Using Agile in DoD Acquisition
- Systems and Software Technology Conference, May 16-19, 2011
  - Presentation: Finding Discipline in an Agile Acquisition Process
- Systems and Software Technology Conference, April 23 -26, 2012
  - Opportunities for New Acquisition Practices: 804 responses
- Pacific NW Quality Conference, Oct 2012
  - Presentation: SW Development & Test as a Service: How and Why
  - Workshop 1: SW Development & Test as a Service: A Natural Path from Agile
  - Workshop 2: SW Development & Test as a Service: Adoption Tips
- Agile East 2012, Nov 2012
  - Presentation: Ready and Fit: Adopting Agile in Constrained Environments



# Agile Myths References

Version One. “Five Myths of Agile Development”, 2010.

[http://pm.versionone.com/whitepaper\\_5myths.html](http://pm.versionone.com/whitepaper_5myths.html)

“Summary of Agile Myths”. <http://stackoverflow.com/questions/1871110/agile-myths-and-misconceptions>

Japikse, P. “Top 30 Agile Myths-Busted”. Telerik Team Pulse.

<http://www.telerik.com/documents/team-productivity-tools/Top30-AgileMyths-Busted.pdf>

Agile Connection. “Top 12 Agile Myths”.

<http://www.agileconnection.com/article/top-twelve-myths-agile-development>

Loffler, M. “7 Agile Myths”. <http://blog.scrumphony.com/2012/06/7-agile-myths/>



# PMI Agile Recommended Reading

*The Program Management Institute has started an Agile certification program. The following is the institute's 2013 list of references:*

*Agile Retrospectives: Making Good Teams Great*  
Esther Derby, Diana Larsen, Ken Schwaber  
ISBN #0977616649

*Agile Software Development: The Cooperative Game – Second Edition*  
Alistair Cockburn  
ISBN #03214827

*The Software Project Manager's Bridge to Agility*  
Michele Sliger, Stacia Broderick  
ISBN #0321502752

*Coaching Agile Teams*  
Lyssa Adkins  
ISBN #0321637704

*Agile Project Management: Creating Innovative Products – Second Edition*  
Jim Highsmith  
ISBN #0321658396

*Becoming Agile: ...In an Imperfect World*  
Greg Smith, Ahmed Sidky  
ISBN #1933988258

*Agile Estimating and Planning*  
Mike Cohn  
ISBN #0131479415

*The Art of Agile Development*  
James Shore  
ISBN #0596527675

*User Stories Applied: For Agile Software Development*  
Mike Cohn  
ISBN #0321205685

*Agile Project Management with Scrum*  
Ken Schwaber  
ISBN #073561993X

*Lean-Agile Software Development: Achieving Enterprise Agility*  
Alan Shalloway, Guy Beaver, James R. Trott  
ISBN #0321532899  
See [www.pmi.org/Certification/New-PMI-Agile-Certification.aspx](http://www.pmi.org/Certification/New-PMI-Agile-Certification.aspx) for updated information.



# Other Resources used in this briefing

Slides 16, 39:

Source: Dr. Paul D. Nielsen Director and Chief Executive Officer of the Software Engineering Institute (SEI), Carnegie Mellon  
Before the United States House of Representatives Defense Acquisition Reform Panel of the Committee on Armed  
Services. July 9, 2009. 8:00 a.m.

[http://democrats.armedservices.house.gov/index.cfm/files/serve?File\\_id=a7a6c258-de85-430c-97f1-a016893b68c7](http://democrats.armedservices.house.gov/index.cfm/files/serve?File_id=a7a6c258-de85-430c-97f1-a016893b68c7)

Slides 16, 39:

<http://www.zdnet.com/blog/projectfailures/10-reasons-for-it-failure/730>

